HPSS Management Guide

High Performance Storage System
Release 7.3

November 2009 *(Revision 1.0)*

HPSS Release 7.3

November 2009 (Revision 1.0)

Table of Contents

List of Tables

Preface

Who Should Read This Book

The HPSS Management Guide is intended as a resource for HPSS administrators. For those performing the initial configuration for a new HPSS system, Chapter 1 provides a configuration roadmap. For both new systems and those upgraded from a previous release, Chapter 1 provides a configuration, operational, and performance checklist which should be consulted before bringing the system into production. The remaining chapters contain the details for configuring, reconfiguring, monitoring, and managing an HPSS system.

Conventions Used in This Book

Example commands that should be typed at a command line will be proceeded by a percent sign (‘**%**’) and be presented in a boldface courier font:

```
% sample command
```

Any text preceded by a pound sign (‘**#**’) should be considered comment lines:

```
# This is a comment
```

Angle brackets (‘<>’) denote a required argument for a command:

```
% sample command <argument>
```

Square brackets (‘**[]**’) denote an optional argument for a command:

```
% sample command [optional argument]
```

Vertical bars (‘|’) denote different choices within an argument:

```
% sample command <argument1 | argument2>
```

A byte is an eight bit data octet. A kilobyte, KB, is 1024 bytes (2^{10} bytes). A megabyte, MB, is 1048576 bytes (2^{20} bytes). A gigabyte, GB, is 1073741824 bytes (2^{30} bytes), a terabyte, TB, is 1099511627776 bytes (2^{40} bytes), and a petabyte, PB, is 1125899906842624 bytes (2^{50} bytes).

Chapter 1. HPSS 7.1 Configuration Overview

1.1. Introduction

This chapter defines the high-level steps necessary to configure, start, and verify correct operation of a new 7.1 HPSS system, whether that system is created from scratch or created by conversion from a 6.2 HPSS system.

To create or modify the HPSS configuration, we recommend that the administrator first be familiar with the information described in the *HPSS Installation Guide*, Chapter 2: *HPSS Basics* and Chapter 3: *HPSS Planning*.

Before performing the procedures described in this chapter, be certain that the appropriate system preparation steps have been performed. See the *HPSS Installation Guide,* Chapter 4: *System Preparation* for more information. For a system created from scratch, be certain that the HPSS installation and infrastructure configuration have been completed. See the *HPSS Installation Guide,* Chapter 5: *HPSS Installation and Infrastructure Configuration* for more information. To convert from a 6.2 system, see the *HPSS Conversion Guide* for HPSS release 7.1.

1.2. Starting the SSM GUI for the First Time

The HPSS system is ready to be configured using SSM once the HPSS software is installed on the node and the HPSS infrastructure components are configured. In order to start the SSM GUI you must first start all infrastructure components and the SSM System Manager as follows:

```
% /opt/hpss/bin/rc.hpss -m start
```

Next you will need to add an SSM Admin user. To do this you will need to invoke the **hpssuser** utility as follows:

```
% /opt/hpss/bin/hpssuser -add hpss -<unix|krb> -ssm
```

The above commands must be done as root!

Once the SSM Admin user has been created, you can invoke the SSM GUI as follows (for hpssgui.pl options, see the hpssgui man page):

```
% /opt/hpss/bin/hpssgui.pl
```

`Note:  `*`This command may be done as an HPSS user.`*

When the SSM GUI is running you can begin to configure the rest of HPSS (servers, devices, etc) as described in the following sections. For more information on SSM, see Chapter 3: *Using SSM* on page 31.

1.3. HPSS Configuration Roadmap (New HPSS Sites)

The following steps summarize the configuration of an HPSS system when creating the 7.1system from scratch (not upgrading from a previous release). It is important that the steps be performed in the order listed. Each step is required unless otherwise indicated. Each step is discussed in more detail in the referenced section.

1. Configure storage subsystems (Section 4.2.2: *Creating a New Storage Subsystem* on page 76)

*Subsystems can be configured only partially at this time. The **Gatekeeper**, **Default COS**, and **Allowed COS** fields will be updated in a later step.*

2. Configure HPSS storage policies
   - Accounting Policy (Section 13.2.1: on page 330)
   - Log Policies (Section 9.2: *Log Policies* on page 295)
   - Location Policy (Section 2.4: *Location Policy* on page 26)
   - Migration Policies (Section 6.4: *Migration Policies* on page 180)
   - Purge Policies (Section 6.5: *Purge Policies* on page 189)
3. Configure HPSS storage characteristics
   - Storage Classes (Section 6.1.1: *Configured Storage Classes* on page 157)
   - Storage Hierarchies (Section 6.2: *Storage Hierarchies* on page 170)
   - Classes of Service (Section 6.3: *Classes of Service* on page 174)
4. Configure HPSS servers (Section 5.1: *Server Configuration* on page 87)
5. Create global configuration (Section 4.1: *Global Configuration Window* on page 72)
6. Configure MVR devices and PVL drives (Section 7.1: *Configure a New Device & Drive* on page 196)
7. Configure file families, if used (Section 6.6: *File Families* on page 193)
8. Update storage subsystem configurations with Gatekeeper and COS information (Section 4.2.4: *Modifying a Storage Subsystem* on page 81 and Section 4.2.3:*Storage Subsystem Configuration Window* on page 76)
9. Create the endpoint map (Section 5.1.3: *Location Server Additional Configuration* on page 99).

1.4. Initial HPSS Startup Roadmap (All Sites)

This section provides instructions for starting the HPSS servers and performing post-startup configuration. For sites which are converting from 6.2, only step 1 may be necessary. For sites configuring a new 7.1 system from scratch, all steps are necessary:

1. Start the HPSS servers (Section 5.2.2: *Starting HPSS Servers* on page 149)
2. Unlock the PVL drives (Section 7.4.2: *Locking a Drive* on page 220)
3. Create HPSS storage space:

   A. Import volumes into HPSS (Section 8.1.1: *Importing Volumes into HPSS* on page 223)

   B. Create storage resources (Section 8.1.2: *Creating Storage Resources* on page 234)

4. Create additional HPSS Users (Section 13.1.1: *Adding HPSS Users* on page 325)
5. Create Filesets and Junctions (Section 10.1: *Filesets & Junctions List* on page 308 and Section 10.5: *Creating a Junction* on page 315)
6. Create HPSS /log Directory

   If log archiving is enabled, using an HPSS namespace tool such as scrub or ftp, create the /log directory in HPSS. This directory must be owned by hpsslog and have permissions rwxr-xr-x.

   The /log directory can be created by the root user using ftp as follows:

```
% ftp <node> <HPSS Port>   # login as root user
ftp> mkdir /log
ftp> quote site chown hpsslog /log
ftp> quote site chmod 755 /log
```

1.5. Additional Configuration Roadmap (All Sites)

This section provides a high level roadmap for additional HPSS configuration.

1. Configure HPSS User Interfaces (Chapter 14: *User Interfaces* on page 339)
2. Set up Backup for DB2 and Other Infrastructure (Chapter 15: *Backup and Recovery* on page 356)
3. Set up High Availability, if desired (*HPSS Installation Guide*, Chapter 3: *HPSS Planning*)
4. Optionally configure support for both authentication mechanisms (*HPSS Installation Guide*, Section 5.9: *Supporting Both Unix and Kerberos Authentication for SSM*)

1.6. Verification Checklists (All Sites)

This section provides a number of checklists regarding configuration, operational and performance issues.

1.6.1. Configuration Checklists

After HPSS is running, the administrator should use the following checklists to verify that HPSS was configured or converted correctly:

Global Configuration

- Verify that a **Default Class of Service** has been selected.
- Verify that a **Root Core Server** has been selected.

Storage Subsystem Configuration

- Verify that a **Default Class of Service** has been selected if desired.
- Verify that a **Gatekeeper** has been selected if gatekeeping or account validation is required.
- Verify that the **COS Name** list has been filled in correctly.

- Verify that a **Core Server** and **Migration Purge Server** have been configured for each storage subsystem.
- Verify that each storage subsystem is accessible by using **lsjunctions** and ensuring that there is at least one junction to the Root fileset of each subsystem. (The root fileset for a given subsystem can be found in the specific configuration for the subsystem's Core Server)

Servers

- Verify that all required HPSS servers are configured and running.
- Verify that all servers are configured with the intended level of logging, whether using their server specific policies or the default policy. Also, verify that all Core Server and Mover log policies have the DEBUG flag turned on to aid in the diagnostics of any future problems.

Devices and Drives

- Verify that all devices/drives are configured and each is assigned to an appropriate PVR/Mover.
- For tape devices, verify that the "**Locate Support**" option is enabled if supported.
- For tape devices, verify that the "**NO-DELAY**" option is enabled if supported by the device.
- For disk devices, verify that the "**Bytes on Device**" and "**Starting Offset**" values are correct.
- Verify that all configured drives are unlocked.

Storage Classes

- Verify that all storage classes are defined and each has sufficient free storage space.
- Verify that each storage class that will be migrated and purged is configured with the appropriate migration and purge policy.
- Verify that no storage class at the lowest level in a hierarchy is configured with a migration or purge policy.
- To support repack and recover of tape volumes, verify that the stripe width of each tape storage class is less than half of the number of available drives of the appropriate drive type.

Storage Hierarchies

- Verify that all storage hierarchies are defined.

Classes of Service (COS)

- Verify that all classes of service are defined.
- Verify that each COS is associated with the appropriate storage hierarchy.
- Verify that the COS is configured to use the characteristics of the hierarchy and the underlying storage classes. In addition, verify that the classes of service have the correct **Minimum File Size** and **Maximum File Size** values. If these sizes overlap, the file placement may be indeterminate when the user creates a file using the size hints. For classes of services which are not to be used as part of standard file placement, set their **Force Selection** flag to ON so that they will only be

chosen if specified by their **COS ID**.

- Verify that classes of service with multiple copies have the **Retry Stage Failures from Secondary Copy** flag enabled.

File Families, Filesets, and Junctions

- Verify that file families and filesets are created according to the site's requirements.
- Verify that each fileset is associated with the appropriate file family and/or COS.
- Verify that each fileset has an associated junction.

User Interfaces

- Verify that the desired HPSS user interfaces (FTP, PFTP, Client API, etc.) are configured (Chapter 14 : *User Interfaces* on page 339).

1.6.2. Operational Checklists

The administrator should follow these checklists to ensure that HPSS is operating properly.

Configured user interfaces

- Create files of various sizes on each defined COS.
- Verify that the files are created in the expected storage class with acceptable transfer rates.
- If necessary, redefine the associated storage class definition to enhance the throughput performance.
- The characteristics fields (**Transfer Rate**, **Latency**, etc.) in the storage class and class of service definition should be updated to reflect actual performance results.
- After the files are created on the correct storage class, verify that the files are created with correct file ownerships and permissions.
- Verify that other file operations (**delete**, **chmod**, **copy**, etc.) work.
- If accounting is configured, verify that the files are created with the correct accounting indices.
- If file families, filesets and junctions are configured, verify that they work as intended.

Devices and Drives

- Ensure that each drive mounts, reads, and writes.

Storage Management

- Verify that migration and purge operations work as intended.
- Start Migration and Purge operations manually to verify that files are migrated and purged correctly.
- Verify that files can be accessed after being migrated/purged.

- Monitor free space from the top level storage class in each hierarchy to verify that the migration and purge policy are maintaining adequate free space.

1.6.3. Performance Checklist

Measure data transfer rates in each COS for:

- Client writes to disk
- Migration from disk to tape
- Staging from tape to disk
- Client reads from disk

Transfer rates should be close to the speed of the underlying hardware. The actual hardware speeds can be obtained from their specifications and by testing directly from the operating system (e.g., using dd to read and write to each device). Keep in mind that transfer performance can be limited by factors external to HPSS. For example, HPSS file read performance may be limited by the performance of the UNIX file system writing the file rather than limits inside HPSS.

Chapter 2. Security and System Access

2.1. Security Services

As of release 6.2, HPSS no longer uses DCE security services. The new approach to security divides services into two APIs, known as mechanisms, each of which has multiple implementations. Configuration files control which implementation of each mechanism is used in the security realm (analogous to a DCE cell) for an HPSS system. Security mechanisms are implemented in shared object libraries and are described to HPSS by a configuration file. HPSS programs that need to use the mechanism dynamically link the library to the program when the program starts.

The first type of mechanism is the **authentication mechanism**. This API is used to acquire credentials and to verify the credentials of clients. Authentication verifies that a client really is who he claims to be.

The second type of mechanism is the **authorization mechanism**. Once a client's identity has been verified, this API is used to obtain the authorization details associated with the client such as uid, gid, group membership, etc., that are used to determine the privileges accorded to the client and the resources to which it has access.

2.1.1. Security Services Configuration

Ordinarily, the configuration files that control HPSS's access to security services are set up either by the installation tool, **mkhpss,** or by the metadata conversion tools. This section is provided purely for reference. Each of the files below is stored by default in **/var/hpss/etc**.

- auth.conf, authz.conf

  These files define which shared libraries provide implementations of the authentication and authorization mechanisms, respectively. They are plain text files that have the same format. Each line is either a comment beginning with # or consists of two fields separated by whitespace: the path to a shared library and the name of the function used to initialize the security interface.

- site.conf

  This file defines security realm options. This is a plain text file in which each line is a comment beginning with # or is made up of the following fields, separated by whitespace:

  ```
  <siteName> <realmName> <realmID> <authzMech> <authzURL>
  ```

  - *<siteName>* - the name of the local security site. This is usually just the realm name in lowercase.
  - *<realmName>* - the name of the local security realm. If using Kerberos authentication, this is the name of the Kerberos realm. For UNIX authentication, it can be any non-empty string. By convention, it is usually the fully qualified hostname.
  - *<realmID>* - the numeric identifier of the local security realm. If using Kerberos authentication and this is a preexisting site going through conversion, this value is the same as the DCE cross cell ID which is a unique number assigned to each site. A new site setting up a new HPSS system will need to contact an HPSS support representative to obtain a unique value.
  - *<authzMech>* - the name of the authorization mechanism to be used by this HPSS system.

This can be "unix" or "ldap".

- *<authzURL>* - a string used by the authorization mechanism to locate the security data for this realm. This should be "unix" for UNIX authorization, and for LDAP it should be an LDAP URL used to locate the entry for the security realm in an LDAP directory.

2.1.2. Security Mechanisms

HPSS 7.1 supports UNIX and Kerberos mechanisms for authentication. It supports LDAP and UNIX mechanisms for authorization.

2.1.2.1. UNIX

UNIX-based mechanisms are provided both for authentication and authorization. These can draw either from the actual UNIX user and group information on the current host or from a separately maintained set of files used only by HPSS. This behavior is controlled by the setting of the variable HPSS_UNIX_USE_SYSTEM_COMMANDS in /var/hpss/etc/env.conf. If this variable is set to any non-empty value other than FALSE, the actual UNIX user and group data will be used. Otherwise, local files created and maintained by the following HPSS utilities will be used. Consult the man pages for each utility for details of its use.

- **hpss_unix_keytab** - used to define "keytab" files that can be used to acquire credentials recognized by the UNIX authentication mechanism.
- **hpss_unix_user** - used to manage users in the HPSS password file (/var/hpss/etc/passwd).
- **hpss_unix_group** - used to manage users in the HPSS groups file (/var/hpss/etc/group).
- **hpss_unix_passwd** - used to change passwords of users in the HPSS password file.
- **hpss_unix_keygen** - used to create a key file containing a hexadecimal key. The key is used during UNIX authentication to encrypt keytab passwords. The encryption provides an extra layer of protection against forged passwords.

Keep in mind that the user and group databases must be kept synchronized across all nodes in an HPSS system. If using the actual UNIX information, this can be accomplished using a service such as NIS. If using the HPSS local files, these must manually be kept in synchronization across HPSS nodes.

2.1.2.2. Kerberos 5

The capability to use MIT Kerberos authentication is provided in HPSS 7.1, however, IBM Service Agreements for HPSS do not provide support for problem isolation nor fixing defects (Level 2 and Level 3 support) in MIT Kerberos. Kerberos maintenance/support must be site-provided.

Kerberos 5 is an option for the authentication mechanism. When this option is used, the local realm name is taken to be the name of a Kerberos realm. The Kerberos security services are used to obtain and verify credentials.

2.1.2.3. LDAP

⚠ *LDAP authorization is not supported by IBM Service Agreements. The following information is provided for sites planning to use LDAP authorization with HPSS 7.1 as a site supported feature.*

An option for the authorization mechanism is to store HPSS security information in an LDAP directory. LDAP (Lightweight Directory Access Protocol) is a standard for providing directory services over a TCP/IP network. A server supporting the LDAP protocol provides a hierarchical view of a centralized repository of data and provides clients with sophisticated search options. The LDAP software supported by the HPSS LDAP authorization mechanism is IBM Tivoli Directory Server (Kerberos plug-in available for AIX only) and OpenLDAP (Kerberos plug-in available for AIX and Linux). One advantage of using the LDAP mechanism over the UNIX mechanism is that LDAP provides a central repository of information that is used by all HPSS nodes; it doesn't have to be manually kept in sync.

The rest of this section deals with how to accomplish various administrative tasks if the LDAP authorization mechanism is used.

2.1.2.3.1. LDAP Administrative Tasks

Working with Principals

- **Creating a principal**

  A principal is an entity with credentials, like a user or a server. The most straightforward way to create a new principal is to use the -add and -ldap options of the **hpssuser** utility. The utility will prompt for any needed information and will drive the **hpss_ldap_admin** utility to create a new principal entry in the LDAP server. To create a new principal directly with the **hpss_ldap_admin** utility, use the following command at the prompt:

  ```
  princ create -uid <uid> -name <name> -gid <gid> -home <home>
  -shell <shell> [-uuid <uuid>]
  ```

  If no UUID is supplied, one will be generated.

- **Deleting a principal**

  Likewise, use the -del and -ldap options of the **hpssuser** utility to delete the named principal from the LDAP server. To delete a named principal directly with the **hpss_ldap_admin** utility, use the following command at the prompt:

  ```
  princ delete [-uid <uid>] [-name <name>] [-gid <gid>]
  [-uuid <uuid>]
  ```

  You may supply any of the arguments listed. This command will delete any principal entries in the LDAP information that have the indicated attributes.

Working with Groups

- **Creating a group**

To create a new group, use the following command at the **hpss_ldap_admin** prompt:

```
group create -gid <gid> -name <name> [-uuid <uuid>]
```

If no UUID is supplied, one will be generated.

- **Deleting a group**

  To delete a group, use the following command at the **hpss_ldap_admin** prompt:

  ```
  group delete [-gid <gid>] [-name <name>] [-uuid <uuid>]
  ```

  You may supply any of the arguments listed. This command will delete any group entries in the LDAP information that have the indicated attributes.

- **Adding a member to a group**

  To add a principal to a group, use the following command at the **hpss_ldap_admin** prompt:

  ```
  group add <principal> [-gid <gid>] [-name <name>] [-uuid <uuid>]
  ```

  You may supply any of the arguments listed to select the group to which the named principal will be added.

- **Removing a member from a group**

  To remove a principal from a group, use the following command at the **hpss_ldap_admin** prompt:

  ```
  group remove <principal> [-gid <gid>] [-name <name>]
  [-uuid <uuid>]
  ```

  You may supply any of the arguments listed to select the group from which the named principal will be removed.

Working with Trusted Foreign Realms

- **Creating a trusted foreign realm**

  To add an entry for a trusted foreign realm, use the following **hpss_ldap_admin** command:

  ```
  trealm create -id <realmID> -mech <mechanism> -name <realmName>
  -url <url>
  ```

  The arguments are as follows

  - -id - the numeric realm ID for the foreign realm
  - -mech - a string identifying the authorization mechanism in use at the foreign realm, such as "unix" or "ldap"
  - -name - the name of the foreign realm, e.g. "SOMEREALM.SOMEDOMAIN.COM"
  - -url - the URL of the security mechanism of the foreign realm. This only matters if the foreign realm is using LDAP as its authorization mechanism. If so, this must be the LDAP URL of the main entry for the security realm in the foreign LDAP server. This should be

obtained from the foreign site's administrator. An example would be: "ldap://theirldapserver.foreign.com/cn=FOREIGNREALM.FOREIGN.COM"

- **Deleting a trusted foreign realm**

  To delete an entry for a trusted foreign realm, use the following **hpss_ldap_admin** command:

  ```
  trealm delete [-id <realmID>] [-name <realmName>]
  ```

  Any of the arguments listed can be supplied to select the trusted realm entry that will be deleted.

2.2. HPSS Server Security ACLs

Beginning with release 6.2, HPSS uses a table of access control information stored in the DB2 configuration database to control access to HPSS servers. This is the AUTHZACL table. HPSS software uses the configured authentication mechanism (e.g. Kerberos) to determine a caller's identity via credentials provided by the caller, then uses the configured authorization mechanism to retrieve the details of the caller that determine the access granted. Once the identity and authorization information have been obtained, each HPSS server grants or denies the caller's request based on the access control list information stored in the database.

The default ACLs for each type of server are as follows:

Core Server:

```
r—-c--- user ${HPSS_PRINCIPAL_FTPD}
rw—c--- user ${HPSS_PRINCIPAL_DMG}
rw-c-dt user ${HPSS_PRINCIPAL_MPS}
r--c--- user ${HPSS_PRINCIPAL_NFSD}
rw-c-d- user ${HPSS_PRINCIPAL_SSM}
r--c--- user ${HPSS_PRINCIPAL_FS}
------t any_other
```

Gatekeeper:

```
rw----- user ${HPSS_PRINCIPAL_CORE}
rw-c--- user ${HPSS_PRINCIPAL_SSM}
r-----t any_other
```

Location Server:

```
r--c--t user ${HPSS_PRINCIPAL_SSM}
r-----t any_other
```

Mover:

```
rw-c--t user ${HPSS_PRINCIPAL_SSM}
r-----t any_other
```

PVL:

```
rw---dt user ${HPSS_PRINCIPAL_CORE}
```

```
rw---dt user ${HPSS_PRINCIPAL_PVR}
rw-c-dt user ${HPSS_PRINCIPAL_SSM}
------t any_other
```

PVR:

```
rw---dt user ${HPSS_PRINCIPAL_PVL}
rw-c--t user ${HPSS_PRINCIPAL_SSM}
------t any_other
```

SSM:

```
rwxcidt user ${HPSS_PRINCIPAL_ADM_USER}
------t any_other
```

All other types:

```
rw-c-dt user ${HPSS_PRINCIPAL_SSM}
------t any_other
```

In most cases, the ACLs created by default for new servers should be adequate. In normal operation, the only ACL that has to be altered is the one for the SSM client interface. This is handled automatically by the -ssm option of the hpssuser utility. If, for some reason, an ACL should need to be modified in some other way, the hpss_server_acl utility can be used. See the hpss_server_acl man page for more information.

2.3. SSM User Security

SSM supports two types of users, administrators and operators:

- **admin**. This security level is normally assigned to an HPSS administrator. Administrators may open all SSM windows and perform all control functions provided by SSM.
- **operator**. This security level is normally assigned to an HPSS operator. Operators may open most SSM windows and can perform all SSM control functions except for HPSS configuration.

Security is applied both at the window level and the field level. A user must have permission to open a window to do anything with it at all. If the user does succeed in opening a window, all items on that window may be viewed. Field level security then determines whether the user can modify fields, push buttons, or otherwise modify the window.

The security level of an SSM user is determined by his entry in the access control information table in the HPSS configuration database. The initial security level for a user is assigned when the SSM user is created by **hpssuser**. Security levels may be viewed and modified with the **hpss_server_acl** utility. See also Section 3.3.2.2: *SSM User Authorization* on page 36.

2.4. Location Policy

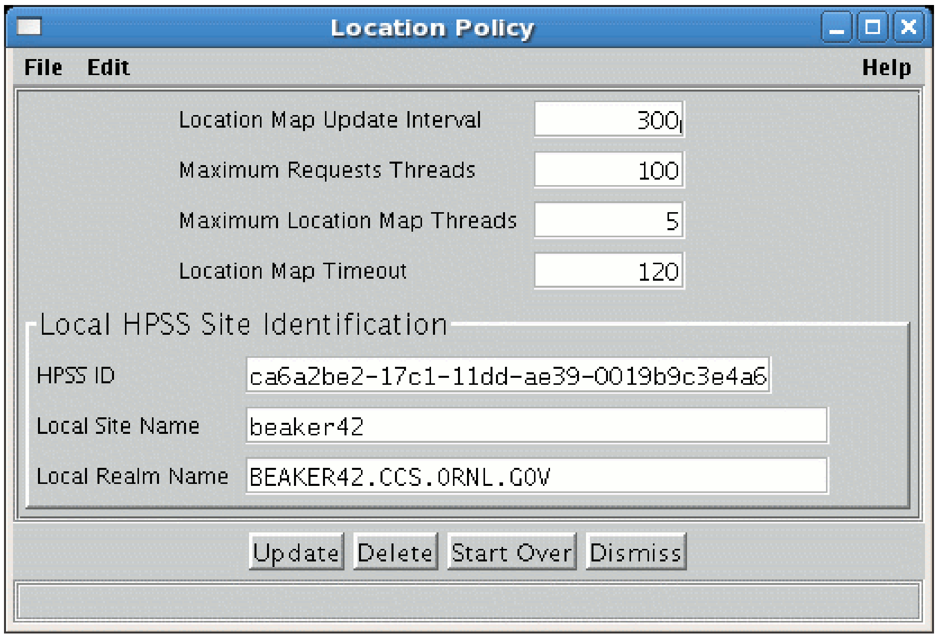
All Location servers in an HPSS installation share a Location Policy. The Location Policy is used by the Location Servers to determine how and how often information should be updated. In general, most of the default values for the policy can be used without change.

2.4.1. Configuring/Updating a Location Policy

The Location Policy can be created and updated using the *Location Policy* window. If the Location Policy does not exist, the fields will be displayed with default values for a new policy. Otherwise, the configured policy will be displayed.

Once a Location Policy is created or updated, it will not be in effect until all local Location Servers are restarted or reinitialized. The Reinitialize button on the *Servers* window can be used to reinitialize a running Location Server.

2.4.2. Location Policy Configuration Window



Location Map Update Interval	300
Maximum Requests Threads	100
Maximum Location Map Threads	5
Location Map Timeout	120

Local HPSS Site Identification

HPSS ID	ca6a2be2-17c1-11dd-ae39-0019b9c3e4a6
Local Site Name	beaker42
Local Realm Name	BEAKER42.CCS.ORNL.GOV

This window allows you to manage the location policy, which is used by HPSS Location Servers. Only one location policy is permitted per HPSS installation.

Once a location policy is created or updated, it will not take effect until all local Location Servers are started or reinitialized. The Reinitialize button on the *Servers* list window (Section 5.1: *Server List* on page 83) can be used to reinitialize a running Location Server.

Field Descriptions

Location Map Update Interval. Interval in seconds that the Location Server rereads the location map.

Advice - If this value is set too low, a load will be put on the database while reading configuration metadata. If set too high, new servers will not be registered in a timely manner. Set this value higher if timeouts are occurring during Location Server communications.

If you have multiple Location Servers, you should consider increasing the update interval since each Location Server obtains information independently and will increase the overall system load.

Maximum Request Threads. The maximum number of concurrent client requests allowed.

Advice - If the Location Server is reporting heavy loads, increase this number. If this number is above 300, consider replicating the Location Server on a different machine. Note if this value is changed, the general configuration thread value (Thread Pool Size) should be adjusted so that its value is always larger than the Maximum Request Threads. See Section 5.1.1.2: Interface Controls on page 92.

Maximum Request Threads *should not normally exceed (****Maximum Location Map Threads*** *+ 400). This is not enforced. If you need more threads than this to handle the load, consider configuring an additional Location Server.*

Maximum Location Map Threads. The maximum number of threads allocated to contact other Location Servers concurrently.

Advice - The actual number of Location Map threads used is ***Maximum Location Map Threads*** *or the number of other HPSS installations to contact, whichever is smaller. This value does not need to be changed unless the system is experiencing timeout problems contacting other Location Servers.*

Location Map Timeout. The maximum amount of time in seconds to wait for a Location Server to return a location map.

Advice - This value should be changed only if the system is experiencing very long delays while contacting another Location Server.

Local HPSS Site Identification:

HPSS ID. The UUID for this HPSS installation.

Local Site Name. The descriptive name of the HPSS installation.

Advice - Pick a name to uniquely describe the HPSS system.

Local Realm Name. The name where the realm containing Location Server path information should be stored.

Advice - All clients will need to know this group name since it is used by them when initializing to contact the Location Server. If the default is not used, ensure that the associated environment variable for this field is changed accordingly for all HPSS interfaces.

2.4.3. Deleting a Location Policy

The Location Policy can be deleted using the *Location Policy* window. Since only one Location Policy may be defined in a system, and it must exist for the system to run, it is better to simply update the policy rather than delete and recreate it. See Section 2.4.2: *Location Policy Configuration Window* on page 27 for more details.

2.5. Restricting user access to HPSS.

System administrators may deny access to HPSS to specific users by including that user in a configuration file that is read by the HPSS Core Server. This file is read by the Core Server at start up time and also read again when the SSM Administrator presses the **Reload List** button on the *Restricted Users* window or whenever the Core Server receives a REINIT request. Any user in this file is denied the usage of the HPSS system completely. To set up this file, you must do the following:

1. Add the HPSS_RESTRICTED_USER_FILE environment variable to /var/hpss/etc/env.conf. Set the value of this variable to the name of the file that will contain the list of restricted users.

   For example:

   HPSS_RESTRICTED_USER_FILE=/var/hpss/etc/restricted_users

2. Edit the file and add the name of the user to the file. The name should be in the form:

   name@realm

   The realm is not required if the user is local to the HPSS realm. For example:

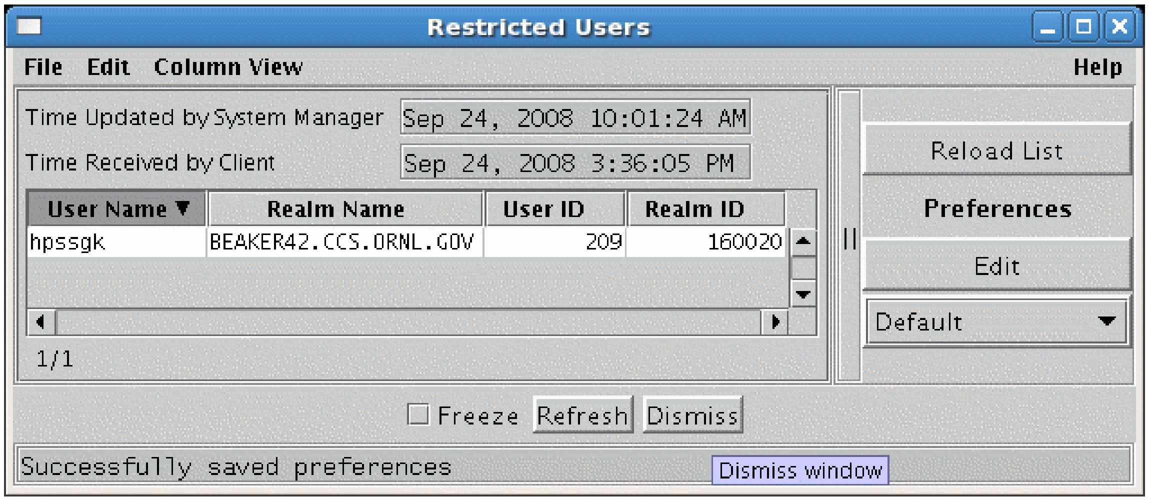
   dsmith@lanl.gov

3. You may put comment lines in the file by beginning the line with a "#".

4. In order for the file to become effective, restart the Core Server, press the **Reload List** button on the *Restricted Users* SSM window or REINIT the Core Server.

   NOTE: The file should be configured on the system where the root Core Server is running; this is the Core Server associated with the Root Name Server. Additionally, if running with multiple storage subsystems on different machines, be sure to configure the HPSS_RESTRICTED_USER_FILE on each machine where a Core Server runs.

2.5.1. Restricted Users Window



Time Updated by System Manager: Sep 24, 2008 10:01:24 AM
Time Received by Client: Sep 24, 2008 3:36:05 PM

User Name	Realm Name	User ID	Realm ID
hpssgk	BEAKER42.CCS.ORNL.GOV	209	160020

1/1

Successfully saved preferences

This window lists all the root Core Server users restricted from HPSS access. To open the window, from the *HPSS Health and Status* window select the Configure menu, and from there select the Restricted Users menu item.

Field Descriptions

Restricted Users list

This is the main portion of the window which displays various information about each restricted user.

User Name. The name of the user that is restricted from HPSS access.

Realm Name. The name of the HPSS realm that encompasses the restricted user.

User ID. The identifier number of the restricted user.

Realm ID. The identifier number which identifies the realm which encompasses the restricted user.

Buttons

Reload List. Issues a request to each Core Server to re-read the restricted user configuration file.

Chapter 3. Using SSM

3.1. The SSM System Manager

3.1.1. Starting the SSM System Manager

Before starting the SSM System Manager (SM), review the SM key environment variables described in the *HPSS Installation Guide,* Section 3.7.10: *Storage System Management*. If the default values are not desired, override them using the **hpss_set_env** utility. See the **hpss_set_env** man page for more information.

To start the SM, invoke the rc.hpss script as follows:

```
% su -
% /opt/hpss/bin/rc.hpss -m start
```

3.1.2. Tuning the System Manager RPC Thread Pool and Request Queue Sizes

Tuning the System Manager RPC Thread Pool and Request Queue sizes can improve the performance of both the System Manager and its clients (hpssgui and hpssadm). It is not necessary, however, to do the tuning when bringing up SSM for the first time. In fact, it can be helpful to postpone the tuning until after the site has a chance to learn its own SSM usage patterns.

The System Manager client interface RPC thread pool size is defined in the **Thread Pool Size** field on the Interface Controls tab of the System Manager's *Core Server Configuration* window (Section 5.1.1.2: *Interface Controls* on page 92). This is the maximum number of RPCs that can be active at any one time for the client interface (i.e. all the hpssgui and hpssadm clients). For the server RPC interface (connections to the SSM System Manager from other HPSS servers), this value is determined by the HPSS_SM_SRV_TPOOL_SIZE environment variable.

The System Manager client interface RPC request queue size is defined in the **Request Queue Size** field on the Interface Controls tab of the System Manager's *Core Server Configuration* window (Section 5.1.1.2: *Interface Controls* on page 92). This is the maximum number of RPC requests from hpssgui and hpssadm clients which can be queued and waiting to become active. For the server RPC interface this value is determined by the HPSS_SM_SRV_QUEUE_SIZE environment variable.

Ideally, if the site runs many clients, the client interface RPC thread pool size should be as large as possible; the default is 100. Testing this value at 300 showed the System Manager memory size more than doubled on AIX from around 32MB to over 70MB. The larger RPC thread pool size makes the System Manager much more responsive to the many clients but also requires it to use more memory.

Experimentation shows that leaving the client interface RPC thread pool size at 100 and leaving the client interface RPC request queue size at its default (600) works pretty well for up to about 40 clients. During further experimentation, setting the client interface RPC request queue size to 1000 resulted in very little effect on memory usage; with 40 clients connected, the client interface RPC request queue used never went above about 500, but the client interface RPC thread pool was constantly filled.

Avoid allowing the client interface RPC thread pool to become full. When this happens, new RPCs will be put into the client interface RPC request queue to wait for a free thread from the thread pool. This makes the client response appear slow, because each RPC request is having to wait its turn in the queue.

To help mitigate this, when the thread pool is full, the System Manager notifies all the threads in the thread pool that are waiting on list updates to return to the client as if they just timed out as normal. This could be as many as 15 threads per client that are awakened and told to return, which makes those threads free to do other work.

If the client interface RPC thread pool is still full (as it could be if, for example, there were 15 threads in the client interface RPC request queue that took over the 15 that were just released), then the System Manager sets the wait time for the new RPCs to 1 second rather than whatever the client requested. This way the RPC won't try to hang around too long.

Realize that once the System Manager gets in this mode (constantly having a full client interface RPC thread pool and having to cut short the thread wait times), the System Manager starts working hard and the CPU usage will start to increase. If you close some windows and/or some clients things should start to stabilize again.

You can see whether the System Manager client interface RPC thread pool has ever been full by looking at the **Maximum Active/Queued RPCs** field in the Client column of the RPC Interface Information group in the *System Manager Statistics* window (Section 3.9.4.1: *System Manager Statistics Window* on page 63). If this number is greater than or equal to the corresponding client interface's Thread Pool Size (default 100), then the thread pool was full at some time during the System Manager execution (although it may not be full currently).

To tell whether the thread pool is currently full, look at the number of Queued RPCs. If Queued RPCs is 0 then the thread pool is not full at the moment.

If Active RPCs is equal to Thread Pool Size then the thread pool for the interface is currently full. Active RPCs should never be greater than Thread Pool Size. When it reaches Thread Pool Size then the new RPCs will be queued and Queued RPCs become greater than 0.

When the thread pool gets full, the System Manager tries harder to clear them out before accepting new ones, so one hopes that if the thread pool fills up, it doesn't stay full for long.

If the site runs with low refresh rates and more than 40 clients, the recommendation is to set the client interface RPC thread pool size to 150 or 200 and the client interface RPC request queue size to 1000 in the System Manager *Server Configuration* window (Section 5.1.1.2: *Interface Controls* on page 92). Otherwise, the default values should work well.

3.1.3. Labeling the System Manager RPC Program Number

Labeling the System Manager RPC program number is not required but can be a useful debugging aid.

The SSM System Manager registers with the RPC portmapper at initialization. As part of this registration, it tells the portmapper its RPC program number. Each HPSS server configuration contains the server's RPC program number. To find the System Manager's program number, open the *Servers* window, select the SSM System Manager, and click the **Configure** button to open the *SSM System Manager Configuration* window. The System Manager's RPC program number is in the **Program Number** field on the Execution Controls tab of this window.

The **rpcinfo** utility with the -p option will list all registered programs, their RPC program numbers, and the port on which they are currently listening for RPCs. When diagnosing SSM problems, it can be useful to run the **rpcinfo** program and search for the System Manager RPC program number in the output, to see whether the System Manager has successfully initialized its rpc interface and to see which port **hpssgui** and **hpssadm** clients must access to reach the System Manager.

This task can be made a bit easier if the System Manager RPC program number is labeled in the portmapper. To do this, add a line for the System Manager in the /etc/rpc file specifying the program number and a convenient rpc service name such as “hpss_ssm” (note that names may not contain embedded spaces). Then this service name will show up in the **rpcinfo** output.

The format of the /etc/rpc file differs slightly across platforms. See the platform specific man pages for the rpc file for details. The **rpcinfo** utility is typically found in either /usr/bin (AIX) or /usr/sbin (Linux).

3.2. Quick Startup of hpssgui

We recommend that **hpssgui** sessions be invoked from the user's desktop computer instead of on the HPSS server machine. **hpssgui** is an application designed to run in the Java environment on the user's desktop computer and to communicate with the remote SSM System Manager. If **hpssgui** is executed on the remote System Manager host, it must run through an X windows session and it may run very slowly in that environment. This is a limitation of Java and networks.

We recognize the value of using the remote X functionality as a quick way to get SSM running, but once your system is up, it is highly recommended that you configure local desktop SSM **hpssgui** clients for all HPSS administrators and operators. Local desktop **hpssgui** configuration is detailed in Section 3.3: *Configuration and Startup of hpssgui and hpssadm* below.

Following are steps for quickly configuring and starting an SSM GUI client:

1. Use the **hpssuser** utility to create an SSM user with admin authority. See Section 3.3.2.1: *The hpssuser Utility* on page 35 and the **hpssuser** man page for more information.
2. On Linux systems, set the JAVA_BIN environment variable to point to the Java runtime binary directory. Set the variable in the environment override file, usually /var/hpss/etc/env.conf. It is usually set to something like /usr/java5/bin. The default setting of $JAVA_BIN is /usr/java5/bin which is the usual location of the java binary directory.
3. The **mkhpss** utility generates the ssm.conf SSM configuration text file when configuring the SM. See the *HPSS Installation Guide,* Section 5.3: *Install HPSS/DB2 and Configure HPSS Infrastructure* for more details. Verify the existence of the $HPSS_PATH_SSM/ssm.conf file.
4. Start the **hpssgui** script:

```
% /opt/hpss/bin/hpssgui.pl
```

- Note that the -m option can be used to specify the desired SSM configuration file to be used. When this option is not specified, hpssgui.pl looks for the ssm.conf configuration file in the current directory, then in the directory defined by the HPSS_PATH_SSM environment variable (usually /var/hpss/ssm). If the script doesn't find a configuration file in either directory, it will use default values to start the client.
- Note that the -d (debug) and -S (log file name) options can be used to capture all levels of **hpssgui** logging in a text file. Bear in mind, however, that this can generate significant amounts of log data. (See the **hpssgui** man page.)

- When you have decided on the hpssgui command line that is best for your installation, it will probably be useful to put the command in a shell script for the convenience of all SSM Administrators and Operators. For example, create a file called "gui" and put the following in it:

```
/opt/hpss/bin/hpssgui.pl \
  -m /my_directory/my_ssm.conf \
  -d \
  -S /tmp/hpssguiSessionLog.$(whoami)
```

Please refer to the **hpssgui** man page for an extensive list of command line options. For example, some sites prefer to set the date format to a USA military format using the *-D "kk:mm:ss dd-MMM-yyyy"* option. Additionally, Section 3.3.3: *SSM Configuration File* below provides a table of variables you can set in the SSM configuration file instead of using command line options; this section also covers all the various files that the **hpssgui** script uses.

3.3. Configuration and Startup of hpssgui and hpssadm

This section describes in detail the procedures for configuring SSM and creating an SSM user account with the proper permissions to start up an **hpssgui** or **hpssadm** session. It also explains how to install the SSM client on the user's desktop (the recommended configuration for hpssgui) and how to deal with special situations such as firewalls.

In the discussion that follows, *authentication* ensures that a user is who they claim to be relative to the system. *Authorization* defines the user's rights and permissions within the system.

Like other components of HPSS, SSM authenticates its users by using either Kerberos or UNIX. Users of the **hpssgui** and **hpssadm** utilities are authenticated to SSM by either a Kerberos principal and a password or by a UNIX username and a password. The System Manager must be configured to use the appropriate authentication and a Kerberos principal or UNIX user account must be created for each SSM user.

Unlike other components of HPSS, SSM does not use LDAP or UNIX to authorize its users. SSM users are authenticated based on their entries in the HPSS DB2 AUTHZACL table. Through this table, SSM supports two levels of SSM client authorization:

admin	This security level is normally assigned to an HPSS administrator. The admin user can view all SSM windows and perform all control functions provided by SSM.
operator	This security level is normally assigned to an HPSS operator. The operator user can view most SSM windows and perform all SSM control functions except for changing the HPSS configuration.

Configuration of an SSM user requires that:

1. The System Manager is configured to accept the desired authentication mechanism.
2. The proper user accounts are created:
   - UNIX or Kerberos accounts are created for the user authentication.

- The proper authorization entries for the user are created in the AUTHZACL table.

3. The proper SSM configuration files are created and installed.

See Section 3.3.1: *Configuring the System Manager Authentication for SSM Clients*, Section 3.3.2: *Creating the SSM User Accounts*, and Section 3.3.3: *SSM Configuration File* for the procedures for these tasks.

See Section 3.3.4: *SSM Help Files (Optiona* on page 42, for instructions on installing the SSM help package.

See Section 3.3.5: *SSM Desktop Client Packaging* on page 42, for instructions for installing **hpssgui** or **hpssadm** on the user's desktop.

See Section 3.3.6: *Using SSM Through a Firewall* on page 44 for advice about using **hpssgui** or **hpssadm** through a network firewall.

3.3.1. Configuring the System Manager Authentication for SSM Clients

The System Manager is configured initially by **mkhpss** for new HPSS systems or by the conversion utilities for upgraded HPSS systems to use the proper authentication mechanism.

If it is necessary later to modify the authentication mechanism for **hpssgui** or **hpssadm** users, or to add an additional mechanism, bring up the *Servers* window, select the System Manager, and press the **Configure** button. On the *System Manager Configuration* window, select the Interface Controls tab. For the SSM Client Interface, make certain the checkbox for the desired Authentication Mechanism, KRB5 or UNIX, is selected. Both mechanisms may be enabled if desired.

Next, select the Security Controls tab. If Kerberos authentication is desired, make certain one of the Authentication Service Configurations is set to use a Mechanism of KRB5, an Authenticator Type of Keytab, and a valid keytab file name for Authenticator (default is /var/hpss/etc/hpss.keytab). If UNIX authentication is desired, make certain one of the Authentication Service Configurations is set to use a Mechanism of UNIX, an Authenticator Type of None, and no Authenticator.

To remove an authentication mechanism from the System Manager, so that no SSM user may be authenticated using that mechanism, reverse the above process. Unselect the mechanism to be removed from the SSM Client Interface on the Interface Controls tab. On the Security Controls tab, change the **Mechanism** and **Authenticator Type** fields of the mechanism to be removed to Not Configured, and change its Authenticator to blank.

See Section 5.1.1.2: *Interface Controls* on page 92, and Section 5.1.1.1: *Security Controls* on page 92, for more information.

3.3.2. Creating the SSM User Accounts

3.3.2.1. The hpssuser Utility

The **hpssuser** utility is the preferred method for creating, modifying or deleting SSM users. It creates the necessary UNIX or Kerberos accounts. It creates an entry in the AUTHZACL table for the user with the proper authorization.

The following is an example of using the **hpssuser** utility to provide administrative access to SSM to user 'john'. In this example, the user already has either a UNIX or Kerberos account.

```
% /opt/hpss/bin/hpssuser -add john -ssm
[ adding ssm user ]
1) admin
2) operator
Choose SSM security level
(type a number or RETURN to cancel):
> 1
[ ssm user added : admin ]
```

After SSM users are added, removed, or modified, the System Manager will automatically discover the change when the user attempts to login. See the **hpssuser** man page for details.

Removing an SSM user or modifying an SSM user's security level won't take effect until that user attempts to start a new session. This means that if an SSM user is removed, any existing SSM sessions for that user will continue to work; access won't be denied until the SSM user attempts to start a new SSM session. Likewise, if the SSM user's security level is changed, any existing sessions for that user will continue to work at the old security level; the new security level access won't be recognized until the SSM user starts a new SSM session).

3.3.2.2. SSM User Authorization

SSM user authorization is set properly by the hpssuser utility with no further modification required. This section explains how the authorization levels are stored internally and how they may be viewed for debugging or modified.

The SSM admin and operator security authorization levels are defined in the AUTHZACL table in the HPSS DB2 database. Each SSM user must have an entry in this table. The permissions supported in the table are:

- **r** – read
- **w** – write
- **x** – execute
- **c** – control
- **i** – insert
- **d** – delete
- **t** – test

SSM administrators must be granted all permissions: rwxcidt. SSM operators must be granted r—c—t permissions. All other permission combinations are not recognized by the SSM server and will be treated as no permissions at all.

The AUTHZACL table may be viewed or updated with the **hpss_server_acl** utility. The **hpssuser** utility program creates and deletes SSM user entries in the AUTHZACL table using the **hpss_server_acl** utility. Normally, there is no need to invoke the **hpss_server_acl** utility directly because it is invoked by the **hpssuser** utility. However, it is a useful tool for examining and modifying the authorization table.

*Access to the **hpss_server_acl** program, **hpssuser** program, to the HPSS DB2 database, and to all HPSS utility programs should be closely guarded. If an operator had permission to run these tools, he could modify the type of authority granted to anyone by SSM. Note that access to the database by many of these tools is controlled by the permissions on the /var/hpss/etc/mm.keytab file.*

Here is an example of using the **hpss_server_acl** utility to set up a client's permissions to be used when communicating with the SSM server. Note that the **default** command should be used only when creating the acl for the first time, as it removes any previous entries for that server and resets all the server's entries to the default values:

```
% /opt/hpss/bin/hpss_server_acl
hsa> acl -t SSM -T ssmclient
hsa> show
hsa> default    # Note: ONLY if creating acl for the first time
hsa> add user <username> <permissions>
hsa> show
hsa> quit
```

If the acl already exists, this command sequence gives user 'bill' operator access:

```
% /opt/hpss/bin/hpss_server_acl
hsa> acl -t SSM -T ssmclient
hsa> show
hsa> add user bill r--c--t
hsa> show
hsa> quit
```

Removing an SSM user or modifying an SSM user's security level won't take effect until that user attempts to start a new session. This means that if an SSM user is removed, any existing SSM sessions for that user will continue to work; access won't be denied until the SSM user attempts to start a new SSM session. Likewise, if the SSM user's security level is changed, any existing sessions for that user will continue to work at the old security level; the new security level access won't be recognized until the SSM user starts a new SSM session).

3.3.2.3. User Keytabs (For Use with hpssadm Only)

A keytab is a file containing a user name and an encrypted password. The keytab file can be used by a utility program to perform authentication without human interaction or the need to store a password in plain text. Only the **hpssadm** utility supports access to SSM with a keytab. Each user who will run the **hpssadm** utility will need access to a keytab. It is recommended that one keytab file per user be created rather than one keytab containing multiple users.

Each keytab file should be readable only by the user for whom it was created. Each host from which the **hpssadm** utility is executed must be secure enough to ensure that the user's keytab file cannot be compromised. An illicit process which gained access to a Kerberos keytab file could gain the user's credentials anywhere in the Kerberos realm; one which gained access to a UNIX keytab file could gain the user's credentials at least on the System Manager host.

Keytabs are created for the user by the **hpssuser** utility when the krb5keytab or unixkeytab authentication type is specified. Keytabs may also be created manually with the **hpss_krb5_keytab** or **hpss_unix_keytab** utility, as described below.

3.3.2.3.1. Keytabs for Kerberos Authentication: hpss_krb5_keytab

The **hpss_krb5_keytab** utility may be used to generate a keytab with Kerberos authentication in the form usable by the **hpssadm** program. See the **hpss_krb5_keytab** man page for details.

The Kerberos keytab is interpreted by the KDC of the Kerberos realm specified by the **hpssadm** utility (see the -k and -u options on the **hpssadm** man page). This must be the same Kerberos realm as that used by the System Manager. This means the **hpss_krb5_keytab** utility must be executed on a host in the same realm as the System Manager.

This example for a user named "joe" on host "pegasus" creates a Kerberos keytab file named "keytab.joe.pegasus":

```
% /opt/hpss/bin/hpss_krb5_keytab
HPSS_ROOT is not set; using /opt/hpss
KRB5_INSTALL_PATH is not set; using /krb5
password:
Your keytab is stored at /tmp/keytab.joe.pegasus
```

Note that under AIX, **hpss_krb5_keytab** will not write to an NFS-mounted filesystem. That's why the utility insists on writing the keytab file in /tmp. Once the keytab is generated, it can be copied and used elsewhere, but care should be taken to keep it secure.

3.3.2.3.2. Keytabs for UNIX Authentication: hpss_unix_keytab

The **hpss_unix_keytab** utility may be used to generate a keytab with UNIX authentication in the form usable by the **hpssadm** program. See the **hpss_unix_keytab** man page for details.

The UNIX keytab is interpreted on the host on which the System Manager runs, not the host on which the **hpssadm** client utility runs. The encrypted password in the keytab must match the encrypted password in the password file on the System Manager host. Therefore, the **hpss_unix_keytab** utility must be executed on the host on which the System Manager runs.

The **hpss_unix_keytab** utility must be able to read the user's encrypted password from the password file. If system password files are being used, this means the utility must be executed as root.

This example for a user named "joe" creates a UNIX keytab file named "joe.keytab.unix":

```
% /opt/hpss/bin/hpss_unix_keytab -f joe.keytab.unix add joe
```

This command copies the encrypted password from the password file into the keytab.

Do not use the -r option of the **hpss_unix_keytab** utility; this places a random password into the keytab file. Do not use the -p option to specify the password; this encrypts the password specified on the command line using a different salt than what was used in the password file, so that the result will not match.

3.3.3. SSM Configuration File

The **hpssgui** and **hpssadm** scripts use the SSM configuration file, ssm.conf for configuration.

The **mkhpss** utility will create the SSM configuration file for the security mechanism supported by SSM. The **mkhpss** utility will store the generated ssm.conf at $HPSS_PATH_SSM; the default location is /var/ hpss/ssm. The configuration file will contain host and site specific variables that the **hpssgui** and **hpssadm** script will read. The variables contain information about:

- SSM hostname
- SSM RPC number
- SSM RPC protection level
- SSM security mechanism
- SSM UNIX Realm [only if using UNIX authentication]

If any of these configuration parameters are modified, the ssm.conf file must be updated and redistributed from the server machine to all of the SSM client machines.

Users can also use their SSM configuration file to manage SSM client parameters instead of using the command line options. The **hpssgui** and **hpssadm** scripts can be directed to use an alternate SSM configuration file with the -m option. The default SSM configuration file contains comments describing each of the available parameters that may be set along with any associated environment variable and command line option. The following table documents these variables and the corresponding command line options:

Table 1. SSM General Options

File Option	*Command Line Option*	*Functionality*
HPSS_SSM_ALARM_RATE	-A	Alarm refresh rate
LOGIN_CONFIG	-C	Full path to login.conf file
HPSS_SSM_DATE_FORMAT	-D	Date format pattern
HPSS_SSM_ALARMS_GET	-G	Number of alarms requested per poll
HPSS_SSM_LIST_RATE	-L	How long **hpssgui**/**hpssadm** waits between polling for lists
HPSS_SSM_ALARMS_DISPLAY	-N	Max number of alarms displayed by **hpssgui**
HPSS_SSM_CLASSPATH	-P	Full path to hpss.jar file
LOG_FILE	-S	Full path for session log file
HPSS_SSM_WAIT_TIME	-W	How long SM waits before returning if object is unchanged
HPSS_SSM_CNTRY_CODE	-c	Country code, internationalization
HPSS_SSM_DEBUG	-d	Debug flag

File Option	*Command Line Option*	*Functionality*
HPSS_SSM_SM_HOST_NAME	-h	System manager hostname
HPSS_SSM_USER_PREF_PATH	-i	Path to ssm preferences
JAVA_BIN	-j	Path to java bin directory
KRB5_CONFIG	-k	Full path to krb5.conf file
HPSS_SSM_LANG_CODE	-l	Language code, internationalization
SSM_CONFIG	-m	Full path to SSM configuration file
HPSS_SSM_SM_PORT_NUM	-n	port number OR RPC number:program number
HPSS_SSM_CLIENT_IP	-p	Client IP address
HPSS_SSM_RPC_PROT_LEVEL	-r	RPC protection level
HPSS_SSM_SEC_MECH	-s	Security mechanism
HPSS_SSM_UNIX_REALM	-u	UNIX Realm

Table 2. HPSSGUI Specific Options

File Option	*Command Line Option*	*Functionality*
HPSSGUI_LOOK_AND_FEEL	-F	Look and feel
HPSSGUI_MO_RATE	-M	How long hpssgui waits between polling for managed objects
HPSSGUI_METAL_THEME	-T	Theme file, for look and feel
HPSSGUI_METAL_BG	-b	Background color
HPSS_SSM_HELP_FILES_PATH	-f	Path to help files
HPSS_SSM_HELP_URL_TYPE	-g	Help path URL type

Table 3. HPSSADM Specific Options

File Option	*Command Line Option*	*Functionality*
HPSSADM_USER_NAME	-U	User name for **hpssadm**
HPSSADM_AUTHENTICATOR	-a	Authenticator (keytab path name)
HPSSADM_BATCH_MODE	-b	Batch mode flag

File Option	***Command Line Option***	***Functionality***
HPSS_AUTHEN_TYPE	-t	Authenticator type

Information on tuning client polling rates for optimal performance is available in the **hpssadm** and **hpssgui** man pages.

Options are specified, in precedence order, by 1) the command line, 2) the user's environment (see the man pages for environment variable names), 3) the SSM configuration file, or 4) internal default values.

3.3.3.1. login.conf

The login.conf file is a login configuration file that specifies the security authentication required for the **hpssgui** and **hpssadm** programs. A copy of the login.conf file is included in the hpss.jar file and should require no site customization. However, a template for the file is provided in /opt/hpss/config/templates/ login.conf.template should the site need to customize the security mecahnisms.

Please see the /opt/hpss/config/templates/login.conf.template file for details.

3.3.3.2. krb5.conf (For Use with Kerberos Authentication Only)

The krb5.conf file is the Kerberos configuration file which allows the client to authenticate to the Kerberos realm. This file is only required if Kerberos authentication is used. The Kerberos installation process generates a default Kerberos configuration file in /etc/krb5.conf.

The following is an example of this file. Realm names, host names, etc. must be customized to operate properly in the user's site environment.

krb5.conf:

```
[logging]
 default = FILE:/var/hpss/log/krb5libs.log
 kdc = FILE:/var/hpss/log/krb5kdc.log
 admin_server = FILE:/var/hpss/log/kadmind.log

[libdefaults]
 ticket_lifetime = 24000
 default_realm = EXAMPLE.COM
 default_keytab_name = /etc/v5srvtab
 default_tkt_enctypes = des-cbc-crc
 default_tgs_enctypes = des-cbc-crc

[realms]
 EXAMPLE.COM = {
  kdc = example.com:88
  admin_server = example.com:749
 }

[domain_realm]
 example.com = EXAMPLE.COM
```

Note that having encryption types other than "des-cbc-crc" first on the "default_tkt_enctypes" and "default_tgs_enctypes" lines can cause authentication failures. Specifically, keytab files generated by the HPSS utility programs will use the first encryption type and only "des-cbc-crc" is known to work in all cases. Other encryption types are known to fail for some OSs and Java implementations. Also, when kinit is used with a keytab file, it only checks the first encryption type listed on the default lines in krb5.conf. If the keytab was generated with a different encryption type, the authentication will fail.

3.3.4. SSM Help Files (Optional)

The SSM Help Files are an HTML version of the *HPSS Management Guide*. Individual sections of this guide are available from the Help menus on the SSM windows.

To access help windows from the **hpssgui**, the Help Files must be accessible from each client machine. We recommend storing these files in a file system shared by the clients so that they don't need to be installed on every SSM client machine. By default, the **hpssgui** script looks for the help files in $HPSS_HELP_FILES_PATH. The default location is /var/hpss/doc and can be overridden by using the -f option.

Help files are distributed with HPSS or can be downloaded from the HPSS web site. They should be installed in the $HPSS_HELP_FILES_PATH location and/or the path specified by the -f option. Refer to the *HPSS Installation Guide*, Section 5.5 *HPSS Documentation & Manual Page Setup* for instructions on how to install the help files. See the **hpssgui** man page for more details.

3.3.5. SSM Desktop Client Packaging

A full installation of HPSS is not needed on machines used only for executing **hpssgui** or **hpssadm**. These machines, referred to here as "SSM client machines", only require the proper version of Java plus a subset of HPSS components.

It is strongly recommended that a desktop configuration be created and installed for each **hpssgui** user. The hpssgui program may run very slowly if it is executed on the System Manager machine and displayed back to the user's desktop via remote X.

There is no advantage to executing the **hpssadm** program on the desktop machine. It will perform just as well when executed remotely as on the desktop. In fact, it is recommended that **hpssadm** be executed on the System Manager machine rather than on the user's desktop since this simplifies the dissemination and protection of the user keytabs. Instructions are included here, however, for packaging the **hpssadm** for sites who have a need to execute it on the desktop.

If the SSM code on the System Manager machine is recompiled , or the System Manager is reconfigured, the client package will no longer work as then it is possible for the hpss.jar file to be out of sync with the System Manager code. **Since each client will have its own copy of the hpss.jar file the hpss.jar file should be redistributed to each client. This can be done by redistributing the entire SSM client package or by just redistributing the hpss.jar file**.

Section 3.3.5.1: *Automatic SSM Client Packaging and Installation*, describes how to use the **hpssuser** utility to package these components. Section 3.3.5.2: *Manual SSM Client Packaging and Installation*, describes how to select and package the components manually.

3.3.5.1. Automatic SSM Client Packaging and Installation

The **hpssuser** utility provides a mechanism for packaging all the necessary client files required to execute the **hpssgui** program on the user's desktop host. Refer to the **hpssuser** man page for more information on generating an SSM Client Package. These files may also be copied manually; see Section 3.3.5.2: *Manual SSM Client Packaging and Installation,* for a list of the required files.

This example creates an SSM Client Package named "ssmclient.tar":

```
%/opt/hpss/bin/hpssuser -ssmclientpkg ssmclient.tar
[ packaging ssm client ]
[ creating ssmclient.tar ]
hpssgui.pl
hpssgui.vbs
hpss.jar
krb5.conf
ssm.conf
[ packaged ssm client in ssmclient.tar ]
```

Once the SSM Client Package has been generated simply FTP the tar file over to the client node and then extract the member files to the desired location.

3.3.5.2. Manual SSM Client Packaging and Installation

This section describes the manual installation of the necessary client files required to execute the **hpssgui** or **hpssadm** program on the user's desktop host. The **hpssuser** utility also provides a mechanism for packaging these files automatically; see Section 3.3.5.1: *Automatic SSM Client Packaging and Installation*.

The desktop machine requires the proper version of Java and the following HPSS files, which should be copied from the host on which the SSM System Manager executes:

• scripts: hpssgui.pl, hpssgui.vbs, hpssadm.pl, or hpssadm.vbs	
• hpss.jar	
• ssm.conf	
• krb5.conf	[if using Kerberos authentication]
• user keytab	[if using **hpssadm**]
• help files	[optional]

These are the default locations of these files on the SSM System Manager host, from which they may be copied:

```
startup scripts /opt/hpss/bin
hpss.jar        /opt/hpss/bin
ssm.conf        /var/hpss/ssm
krb5.conf       /etc/krb5.conf
keytab file     /var/hpss/ssm/keytab.USERNAME
help files      /var/hpss/doc
```

These files may be installed in any location on the SSM client machines. The user must have at least read access to the files.

The SSM startup scripts hpssgui.pl, hpssgui.vbs, hpssadm.pl, and hpssadm.vbs provide the user with a command line mechanism for starting the SSM client. The hpssgui.pl script is a Perl script for starting the SSM Graphical User Interface and the hpssadm.pl script is a Perl script for starting the SSM Command Line User Interface. These scripts work on AIX, Linux, or Windows platforms so long as Perl is installed on the host. The hpssgui.vbs script is a Visual Basic script for starting the Graphical User Interface and the hpssadm.vbs script is a Visual Basic Script for starting the SSM Command Line User Interface. These scripts work only on Windows platforms.

These scripts depend on the ability to read the other files in the package. See the **hpssgui** and **hpssadm** man pages for details**.**

The hpss.jar file contains the **hpssadm** and **hpssgui** program files. This is stored on the server machine under $HPSS_PATH_BIN; the default location is /opt/hpss/bin. If the SSM source code on the server machine is recompiled, the hpss.jar file must be redistributed to all of the SSM client machines.

The keytab is used only by the **hpssadm** program. See Section 3.3.2.3: *User Keytabs (For Use with hpssadm Only)* on page 37*,* for details.

See Section 3.3.4: *SSM Help Files (Optiona* on page 42, for a description of the Help Files.

A writable directory is required for **hpssgui** or **hpssadm** session logs, if these are desired. The session log is an ASCII file that stores messages generated by the **hpssadm** or **hpssgui** programs. By default, the **hpssgui**/**hpssadm** scripts do not create session logs, but it is strongly encouraged that this capability be enabled by using the -S <location> option when running the script. The recommended location is /tmp on UNIX-like systems or c:\tmp on Windows systems. See the **hpssgui** and **hpssadm** man pages for more information on creating a session log. Having the session log available helps when debugging problems with the SSM client applications. It is the first thing that the SSM developers will ask for when someone is having problems with the **hpssgui** and/or **hpssadm**.

3.3.6. Using SSM Through a Firewall

3.3.6.1. The Firewall Problem

hpssgui and **hpssadm** require the use of several network ports which may be blocked if the client and System Manager are on opposite sides of a network firewall. Up to three ports may be affected:

- **hpssgui** and **hpssadm** must be able to access the port upon which the System Manager listens for requests.
- If the System Manager follows the default behavior of letting the portmapper select this port, then **hpssgui** and **hpssadm** also need access to port 111 in order to ask the portmapper where the System Manager is listening.
- If Kerberos authentication is used, then **hpssgui** and **hpssadm** additionally need access to port 88 to communicate with the KDC.

3.3.6.2. Solutions for Operating Through a Firewall

SSM can operate through a firewall in three different ways:

- The **hpssgui** and **hpssadm** can use ports exempted by the network administrator as firewall exceptions. See the -n option described in the **hpssgui** and **hpssadm** man pages.
- The **hpssgui** and **hpssadm** can contact the System Manager across a Virtual Private Network connection (VPN). See the -p and -h options described in the **hpssgui** and **hpssadm** man pages.
- The **hpssgui** and **hpssadm** can contact the System Manager across an ssh tunnel. See the instructions for tunneling in the **hpssgui** man page.

The firewall exception is the simplest of these. However, security organizations are not always willing to grant exceptions.

The vpn option is usually simple and transparent regardless of how many ports are needed, but requires the site to support vpn. The site must also allow the vpn users access to the ports listed in Section 3.3.6.1 *The Firewall Problem* on page 44; not all sites do.

The ssh tunneling option has the advantage that it can be used almost anywhere at no cost. It has the disadvantage that the tunnel essentially creates its own firewall exception. Some security organizations would rather know about any applications coming through the firewall and what ports they are using rather than have users create exceptions themselves without the awareness of security personnel. A second disadvantage of tunneling is that if a particular client machine is compromised, any tunnels open on that client could also be compromised. The client machine may become a point of vulnerability and access to the other machines behind the firewall. A third disadvantage is that tunneling can be complex to set up, requiring slight or significant variations at every site.

The firewall and tunneling options both benefit from reducing the number of ports required:

- The need for port 111 can be eliminated by making the System Manager listen on a fixed port. To do this, set the HPSS_SSM_SERVER_LISTEN_PORT environment variable to the desired port and restart the System Manager. Then use the -n option with the **hpssgui** and **hpssadm** startup scripts to specify this port.
- The need for port 88 can be eliminated only by avoiding Kerberos and using UNIX authentication.
- There is no way to eliminate the need for the port on which the System Manager listens.

3.3.6.3. Example: Using hpssgui Through a Firewall

Here is an example of how a particular site set up their hpssgui SSM client sessions using krb5 authentication outside a firewall. Many of the items are site specific so modifications will need to be made to suit each site's specific needs. Where this procedure would differ for a site using Unix authentication, the Unix instructions are also included.

At this site, vpn users were not allowed access to all the ports listed in Section 3.3.6.1 *The Firewall Problem* on page 44 so they had to use a combination of vpn and ssh tunneling.

- Create a directory on the client machine to hold the SSM client files. It is recommended that a separate directory be created for each server hostname that the client will contact.

- Verify that the proper version of Java is installed. Add the Java bin directory to the user's $PATH, or use the -j switch in the hpssgui script, or set JAVA_BIN in the user's ssm.conf file. Java can be downloaded from http://www.java.com.
- Obtain files from the server machine:
  - Obtain the preferred **hpssgui** script for the client system from /opt/hpss/bin on the server machine and place it in the directory created on the client machine (see Section 3.3.5: *SSM Desktop Client Packaging* on page 42). There are several script options. Only one version of the script is needed:
    - **hpssgui.pl** which is written in Perl and can be used on any system that has Perl installed. This is true for any major UNIX operating systems as well as MacOS. For Windows users, Perl must be installed to use this version of the script. Users can easily obtain this from the web. A good Perl distribution for Windows is available at http:/www.activestate.com.
    - **hpssgui.vbs** is a Visual Basic Script version for Windows users. This version requires no prerequisite software.
  - Obtain the ssm.conf file from /var/hpss/ssm on the server machine and place it in the directory where the **hpssgui** script resides. Alternately, specify the file to the **hpssgui** script with the -m option, if desired.
  - Obtain the hpss.jar file from /opt/hpss/bin on the server machine and place it in the directory where the **hpssgui** script resides. If FTP is used to copy the file, make sure the copy is done in binary mode. If the file is installed in a different directory, specify it to the **hpssgui** script with the -P option, or by using configuration file settings or the appropriate environment variable (see the **hpssgui** man page).
- If Kerberos authentication is used, be sure to get the krb5.conf file that resides on the SSM server. This file should be located at /etc/krb5.conf. Place this file on the client machine in the directory where the **hpssgui** script resides. Alternately, specify this file to the **hpssgui** script with the -k option. Verify that UDP port 88 on the SSM Server machine is accessible; if not, then **hpssgui** will fail.
- To get access to ports inside the firewall, we can use a vpn connection or one or more ssh tunnels.
  - Using a vpn connection will make it appear that we are inside the firewall. In this case, no tunnels are needed. If the firewall does not permit ssh connections, ssh tunnels cannot be used. Set up the vpn connection on the client machine.
  - If using one or more ssh tunnels is preferred, on the SSM server machine, set the HPSS_SSM_SERVER_LISTEN_PORT environment variable to a specific port (e.g. 49999). Restart the System Manager so that it will recognize this variable.

    On the client machine, set up an ssh tunnel where 49999 corresponds to the HPSS_SSM_SERVER_LISTEN_PORT, the user name is joe and the SSM Server machine is "example.com".

    ```
    % ssh -N -f -L 49999:localhost:49999 joe@example.com
    ```

If access through the firewall is needed for other ports (eg., the Kerberos kdc), set up a separate tunnel for each port the firewall does not allow through.

- On the client machine, run the GUI:
  - For Kerberos authentication:

    ```
    % hpssgui.pl -S hpssgui.sessionlog -k krb5.conf -n 49999 -h
    localhost
    ```

  - For UNIX authentication:

    ```
    % hpssgui.pl -S hpssgui.sessionlog -s unix -u example.com -n
    49999 -h localhost
    ```

The *HPSS Login* window should open on the client machine for the user to log in. If it doesn't, then retry the last step, running the GUI, using the -d option for debug output and the -S option to log output to a session log file. This file will provide some information about what is going wrong.

3.4. Multiple SSM Sessions

Multiple concurrent sessions of the graphical user interface and/or command line utility can be executed by the same user with no special modifications. Each session should specify a unique session log file.

3.5. SSM Window Conventions

This section lists conventions used by SSM and Java, on which SSM is based. The following list does not cover all features of all windows; it only describes the most important points.

- Lists may be sorted by any column. Click on the column header of the desired column to sort the list by the items in that column. The column header will become highlighted and will display an up or down arrow to indicate the direction of the sort. Click the column header a second time to change the direction of the sort.
- List tables have a field that shows the number of displayed and total items in the list in the format X/Y where X is the number of items displayed and Y is the total number of items in the list. The field is left justified under the table. The X and Y values will differ if preferences are set to filter some items out of the list.
- The button panel to the right of the list can be hidden or displayed by clicking the tall, thin button between the list and button panel labeled '||'. If the button is pressed when the panel is displayed, the button panel will hide, allowing more space for the list. The button panel may be re-displayed by pressing the '||' button again.
- Colors and fonts are used consistently from window to window. They may differ from platform to platform because the default Java Look and Feel settings vary between platforms.

  The **hpssgui** script accepts the following flag parameters in order to control the graphical user interface's look and feel:

  - -F "Look and Feel"
    - Valid values: windows, mac, motif, metal, gtk. Select the Look and Feel that is

applicable to the platform on which the graphical user interface is running. Custom Look and Feels are also available at http://www.javootoo.com

- -b "background color"
    - The only Look and Feel that supports color settings and themes is the metal Look and Feel. The color may be set by using the color name or hexadecimal Red/Green/Blue value. Here are some examples:

Name	Hexadecimal value
red	0xff0000
green	0x00ff00
blue	0x0000ff
cyan	0x00ffff
yellow	0xffff00
magenta	0xff00ff

- -T "theme file"
    - The theme setting is only applicable when used with the metal Look and Feel. There are eight color parameters that can be used to customize the look of HPSS windows: three primary colors, three secondary colors, black and white. The color settings may be stored in a theme file using the following syntax:
        - hpss.ssm.ssmuser.hpssgui.CustomMetalTheme.primary1=COLOR
        - hpss.ssm.ssmuser.hpssgui.CustomMetalTheme.primary2=COLOR
        - hpss.ssm.ssmuser.hpssgui.CustomMetalTheme.primary3=COLOR
        - hpss.ssm.ssmuser.hpssgui.CustomMetalTheme.secondary1=COLOR
        - hpss.ssm.ssmuser.hpssgui.CustomMetalTheme.secondary2=COLOR
        - hpss.ssm.ssmuser.hpssgui.CustomMetalTheme.secondary3=COLOR
        - hpss.ssm.ssmuser.hpssgui.CustomMetalTheme.black=COLOR
        - hpss.ssm.ssmuser.hpssgui.CustomMetalTheme.white=COLOR
    - COLOR should be specified using the color name or Red/Green/Blue hexadecimal value (see the example under the -b flag above).
    - If the theme file location is not specified on the command line, the default value used is ${HOME}/hpss-ssm-prefs/DefaultTheme.
- Buttons may be "disabled" when the current state of the window does not allow an operation to be performed. In this state, a button is visible but its label text is grayed out and clicking it has no effect. The disabled state occurs when the operation is not supported for the selected item or the SSM user does not have sufficient authority to perform the operation
- A "text" field is any field which displays alphanumeric text or numeric data. This does not

include “static” text painted on the window background or labels on things like buttons. Text fields may appear as single or multiple lines and they may be “enterable” (the displayed data can be altered) or “non-enterable” (the displayed data cannot be changed directly).

- Non-enterable text fields have gray backgrounds. A particular field may be enterable under one circumstance but non-enterable under another; for example, a server configuration window's Server ID field is enterable during server creation but may not be changed when modifying a pre-existing configuration record. Additionally, a field is non-enterable when the user does not have sufficient authority to modify the field.
- Enterable text fields have white backgrounds. In most cases, when data in the current field is modified and the field loses focus (the cursor leaves the field), a floppy disk icon will be displayed next to the field to give a visual cue that the field has been changed and that the changes have not been saved. When all changes are made, the user can submit the modifications by pressing one of the window’s operation buttons.
- Some enterable text fields are wider than they appear. As typing proceeds and the cursor reaches the right-most edge of the field, the text automatically scrolls to the left and allows further data entry until the actual size of the field has been reached. Scroll back and forth within the field using the left and right cursor keys to view the contents.
- Some text fields which accept integer values can also accept numeric abbreviations such as “KB”, “MB”, “GB”, “TB”, or “XB” to specify kilobytes, megabytes, gigabytes, terabytes, or exabytes, respectively. Character case is ignored. For example, entering "1024" will yield the same results as entering "1kb". The entered value must fall with the acceptable numeric ranges for the specified field.
- Some text fields which accept integer values can accept the values in decimal, octal, or hexadecimal form. For these fields, values which begin with an 'x' or '0x' will be interpreted as hexadecimal and values which begin with a zero '0' (but not '0x') will be interpreted as octal. All other values will be interpreted as decimal.
- A combo box is a non-enterable text field inside a button-like box with a small arrow on the right side. Clicking on the box will pop up a list of items. Selecting a list item will replace the displayed contents of the combo box's text field. Alternately, the list can be dismissed by clicking the mouse anywhere outside of the popup list and the displayed contents will remain unchanged.
- A checkbox is a field containing a box graphic followed by a label. The box may be hollow, ☐, indicating that the item is not selected. It may be filled in, ■, or contain a check mark, ☑, indicating that the item is selected. Clicking on an enterable check box toggles the state of the selected item. When the state of a check box cannot be modified, it will appear gray in color.
- A radio button is a field containing a circle followed by a label. The circle may be hollow, ○, indicating that the item is not selected or may have a solid interior, ●, indicating that the item is selected. Radio buttons are displayed in groups of two or more items. Only one item within the group can be selected; selecting one button in a group will cause all other buttons in the group to become unselected. When the state of the radio buttons cannot be modified, they will appear gray in color.
- An enterable field containing a cursor is said to have “input focus”. If an enterable text field has input focus, typing on the keyboard will enter characters into the field.

- Select/cut/copy/paste operations can be performed on enterable text fields; on non-enterable fields, only select and copy operations can be performed.
- In some cases, modifying a field value or pressing a button causes the action to be performed immediately. A confirmation window will pop up to inform the user that all changes made to the data window will be processed if the user wishes to continue. If the user selects 'No' on the confirmation window, the request will not be processed and any field modifications to the window will continue to be displayed. Some examples are changes to the **Administrative State** field, pressing the Gatekeeper's **Read Site Policy** button, and selecting an entry from the **MPS Storage Class Information Control** combo box.

3.6. Common Window Elements

Certain SSM buttons and toggle boxes have the same behavior on all SSM windows. Descriptions for these common elements are given below and are not repeated for each window:

- **Time Created by System Manager** field - The last time the System Manager created the structure for this window.
- **Time Updated by System Manager** field - The last time the System Manager updated the data for this window.
- **Time Received by Client** field - The last time the SSM client received an update for this window from the System Manager.
- **Dismiss** button - Closes the current SSM window.
- **Add** button – The **Add** button is displayed on configuration windows when a new configuration record is being created. After the configuration fields are appropriately completed, click the **Add** button to save the data and create the new record. When the Add operation is not permitted, the **Add** button will not be displayed or will appear gray in color.
- **Update** button – The **Update** button is displayed on configuration windows when an existing record is being modified. After the configuration's fields have been modified, click the **Update** button to save the modifications. When the update operation is not permitted, the **Update** button will not be displayed or will appear gray in color.
- **Delete** button – The **Delete** button is displayed on configuration windows of existing records. Click the **Delete** button only when the current record is no longer needed and any dependent records have also been deleted. When the Delete operation is not permitted, the **Delete** button will not be displayed or will appear gray in color.
- **Start Over** button - Resets the current values in a configuration window to the values used when the window was first opened.
- **Start New** button - Replace the contents of the current configuration window with a new configuration of the same type as the one being viewed. The new configuration's initial values will contain defaults.
- **Clone (partial)** button - Replace the contents of the current window with a new configuration using some of the current configuration's field values.
- **Clone (full)** button - Replace the contents of the current window with a new configuration using

all of the current configuration's field values.

- **Freeze** - A checkbox that, while checked, suspends the automatic updates made to an SSM window. This allows reviewing information at the frozen point in time. Unchecking the checkbox will reactivate normal update behavior.
- **Refresh** button - Requests an immediate update of the displayed information. This can be useful if the user does not wish to wait for an automatic update to occur.
- **Preferences** button and combo box:
  - **Preferences Edit** button - Clicking the **Edit** button opens a configuration window from which certain display characteristics of the parent window can be modified and saved in a preference record. New preference records can be created by saving the preference record with a new name.
  - **Preferences** combo box - Click on the Preference combo box to view a list of available preference records used to control the information displayed on the window. The preference record can be modified by either selecting another preference record or by modifying the current preference record. See the **Edit** button description above.

- **Status Bar** - A non-enterable text field along the bottom of all SSM data windows. Status lines display messages concerning the state of the window and the progress of operations started from the window. To view status messages that are longer than the length of the status bar, either stretch the window horizontally or mouse-over the status message to see the full text displayed as a tool tip. Alternately, the user can view status messages in the session log file. When the status bar has had messages written to it, the most recent messages can be viewed in the status bar's tooltip. If there are status messages to view, rolling the mouse over the status bar without clicking gives a tooltip that says, "Click mouse in status bar to view messages". If there are no status messages then the tooltip says, "No status messages". This message stays up for about 4 seconds or until the user moves the mouse out of the status bar area. To view up to the last 30 messages that have been written to the status bar, click on the status bar. The tooltip that results will show up to the last 30 messages and will remain visible for 10 minutes or until the mouse is moved out of the status bar.
- **File** menu - All SSM data windows have the File menu. The File menu consists of menu options for controlling the window's location, the user's session, and printing. The File menu offers the following options: Cascade, Page Setup, Print, Close All or Close, Logoff, and Exit. The Cascade, Close All, Logoff, and Exit menu options are only available on the *HPSS Health and Status* window.
  - **Page Setup** - SSM uses Java's print facility to create a dialog box enabling the user to enter directions to be sent to the printer. The *Page Setup* dialog box can be used to specify print media, page orientation, and margin characteristics. The *Page Setup* dialog box can also be accessed via the *Print* dialog box (see below). The Page Setup menu item is available on all SSM windows.
  - **Print** - The Print dialog box is used to set printing options. The print options that are available are platform dependent and therefore may vary. Some print options that can be configured include selecting a printer, setting page size and margins, and specifying the number of copies and pages to print. The Print menu item is available on all SSM windows.
  - **Close** - The Close menu option is used to close the currently selected window. The Close

menu item is available on all SSM windows.

- **Edit** menu - The Edit Menu is located on all SSM data windows. From each Edit Menu, the user can access **Cut**, **Copy** and **Paste** functions which enable the user to remove data from text fields or transfer data among them. Editable text fields can be updated. Non-editable text fields can be copied, but not changed. Field labels cannot be copied.

  Most windowing systems provide keyboard shortcuts for the Cut, Copy, and Paste commands. A typical set of keyboard shortcuts is Ctrl-C for Copy, Ctrl-X for Cut, and Ctrl-V for Paste, but details may vary from system to system. Cut or Copied text can be Pasted into other applications using the keyboard shortcuts.

  - To delete data from a text field - Highlight the characters to be removed and select Cut.
  - To move data from one field to another - Highlight the characters to be moved and select Cut. Then position the cursor where the data should be placed and select Paste.
  - To copy data from one field to another - Highlight the characters to be copied and select Copy. Then position the cursor where the data should be placed and select Paste.

- **Column View** menu – The Column View menu only appears on SSM windows that display an SSM table. An entry for each column in the table appears in the drop down list along with a corresponding checkbox. If the checkbox is selected, then the column will appear in the window's table; otherwise the column will be hidden. Clicking on the checkbox will toggle the hidden or viewable state of the column.
- **Help** menu - All SSM windows have a Help menu. See Section 3.7: *Help Menu Overview* below for detailed information on SSM help.

3.7. Help Menu Overview

The Help Menu provides access to online help that is pertinent to the window being displayed. The Help menu is available on all SSM data windows but is not available on informational windows such as error messages and confirmation windows. After selecting Help, the menu will expand to list the help topics that are available for the current window. Selection of a window-related help topic will open an HTML file and jump to the corresponding topic section. Selection of the *HPSS Management Guide* on a help menu will take the user to the table of contents of the Management Guide.

The *HPSS Management Guide* , along with Chapter 1 of the *HPSS Error Manual,* is the main source for diagnosing and solving problems.

In order to access SSM Help , the help files must be installed and accessible to the graphical user interface. For information on obtaining and installing the SSM Help files, see the *HPSS Installation Guide* section 5.1.2: *Software Installation Packages.*

The SSM Help facility uses two environment variables, HPSS_HELP_URL_TYPE and HPSS_HELP_FILES_PATH, to determine the location of the SSM Help files. The HPSS_HELP_URL_TYPE environment variable specifies the type of URL to aid the browser in locating the help files. Valid URL types are 'https:', 'http:', or 'file:'. The default value for the HPSS_HELP_URL_TYPE is 'file:'. The HPSS_HELP_FILES_PATH environment variable specifies the location of the installation directory for the SSM Help files. The SSM Help files must exist in HTML format for the graphical user interface to display them. The default value for the HPSS_HELP_FILES_PATH environment variable is '/var/hpss/doc'. The values of these environment variables may be overridden.

3.8. Monitor, Operations and Configure Menus Overview

The Monitor, Operations and Configure menus are used by the System Manager to monitor, control and configure HPSS. They are available only from the *HPSS Health and Status* window. This section provides a brief description on each submenu option listed under the Monitor, Operations and Configure menus. See related sections for more detailed information on the window that gets opened after selecting the menu option.

3.8.1. Monitor Menu

The Monitor menu contains submenu items that are commonly used by the System Administrator to monitor the status of HPSS. The menu is organized with the submenu items that open SSM list windows at the top and the submenu items that open other window types at the bottom.

Alarms & Events. Opens the *Alarms and Events* window which displays HPSS alarm and event messages. Detailed information on each alarm or event can be viewed by selecting an entry from the list and pressing the **Alarm/Event Info** button.

Devices & Drives. Opens the *Devices and Drives* window which displays information on all configured Mover devices and PVL drives. Devices and Drives may be created, deleted, viewed, locked, unlocked, dismounted and mark repaired from this window.

PVL Jobs. Opens the *PVL Job Queue* window which displays all active PVL jobs. Use this window to get more detailed information on a job or to cancel a job.

Servers. Opens the *Servers* window which displays information on all configured HPSS servers. This window can also be used to perform server related operations such as configuration, startup, shutdown, and viewing server status.

Storage Classes, Active. Opens the *Active Storage Classes* list window which displays information for all storage classes which have been assigned to a Migration Purge Server and assigned storage resources. Migration, purge, repack and reclaim operations can be initiated from this window. This window also provides access to information for each storage class such as the managed object information, configuration record, migration policy and purge policy. The Migration Purge Server must be running in order for its storage classes to be active and show up in this list.

RTM Summary. Opens the *RTM Summary List* window which displays summary information for the active Real-Time Monitor (RTM) records. RTM records are maintained by the Core, Gatekeeper and Mover components of HPSS.

Filesets & Junctions. Opens the *Filesets & Junctions List* window which displays information for the filesets and junctions that are configured in the HPSS system. Filesets and junctions can be created and deleted and details for filesets can viewed from this window.

Tape Requests. This submenu lists the different tape request list window types.

- **Check-In.** Opens the *Check-In Requests* window which displays all current requests for tape check-ins.
- **Mount.** Opens the *Mount Requests* window which displays all current requests for tape mounts.

Accounting Status. Opens the *Subsystem* list window where the **Accounting Status** and **Start Accounting** buttons can be found.

Log Files Information. Opens the *Log Files Information* window to display information for the HPSS log files such as the log file's size and state.

Lookup HPSS Objects. This submenu lists the type of objects which can be looked up by specifying the object's identifying information.

- **Cartridges & Volumes.** Opens the *Lookup Cartridges and Volumes* window allowing identification of a cartridge or volume by name. A choice can then be made between viewing PVL volume information, PVR cartridge information, or Core Server volume information for the specified cartridge or volume.
- **Files & Directories.** Opens the *Lookup Files and Directories* window into which a pathname can be entered. The pathname can identify a file, directory or junction. From this window, click the **Show File/Directory** button to display detailed information about the file, directory or junction.
- **Objects by SOID.** Opens the *Lookup Object by SOID* window where an HPSS object can be specified by entering its HPSS Storage Object ID (HPSS SOID). This window provides access to bitfile and virtual volume information.

SSM Information. This submenu lists the options available for viewing statistics for the System Manager and the user client session.

- **System Manager Statistics.** Opens the *SSM System Manager Statistics* window to view statistics for the System Manager such as the number of RPC calls, notifications and messages that were processed.
- **User Session Information.** Opens the *User Session Information* window to display the user's login name, authority and statistics regarding the user's session.

3.8.2. Operations Menu

Accounting Report. Opens the *Subsystems* list window where a subsystem can be highlighted and the **Start Accounting** button can be selected to obtain an accounting report.

Drive Dismount. Opens the *Devices and Drives* list window where the **Dismount Drive** button is located.

PVL Job Cancellation. Opens the *PVL Job Queue* window from which PVL jobs may be selected and canceled.

Resources. This submenu lists the operations that can be performed on disk and tape cartridges and volumes.

- **Import Disk Volumes.** Opens the *Import Disk Volumes* window where a list of disk volume labels can be entered and an import request can be submitted.
- **Import Tape Volumes.** Opens the *Import Tape Volumes* window where a list of tape volume labels can be entered and an import request can be submitted.
- **Create Disk Resources.** Opens the *Create Disk Resources* window where a list of disk

volume labels can be entered and a request to add the disks to a storage class can be submitted.

- **Create Tape Resources.** Opens the *Create Tape Resources* window where a list of tape volume labels can be entered and a request to add the tapes to a storage class can be submitted.
- **Delete Resources.** Opens the *Delete Resources* window allowing deletion of existing tape or disk storage resources.
- **Export Volumes.** Opens the *Export Volumes* window which exports tape cartridges and disk volumes, making them unavailable to HPSS.
- **Move Cartridges.** Opens the *Move Cartridges To New PVR* window allowing ownership of tape cartridges to be transferred between PVRs.
- **Migrate/Purge Data.** Opens the *Active Storage Classes* window. From this window, a storage class may be highlighted and a migration or purge can be started by selecting the corresponding button.
- **Repack/Reclaim Tapes.** Opens the *Active Storage Classes* window where a storage class may be highlighted and the **Repack Volumes** or **Reclaim Volumes** button selected to perform the operation.

Ping System Manager. Selecting this submenu option tests the connectivity between the GUI and the System Manager. If the ping is successful, nothing will happen. If the ping is unsuccessful, an error message will be displayed.

Shutdown. This submenu provides a quick way to send a shutdown request to any server other than the Startup Daemon. If you want to shutdown a particular server or set of servers, use the **Shutdown** or **Force Halt** buttons on the *Servers* list window. The System Manager cannot be shutdown via the *Servers* list window. The Startup Daemon cannot be shutdown at all using SSM.

- **All Non-SSM Servers –** Selecting this option sends a shutdown command to all servers other than the System Manager and Startup Daemon. Note that some servers may take a few minutes to shutdown. To restart the servers, select the servers in the *Servers* list window and press the **Start** button.
- **System Manager –** Selecting this option sends a shutdown command to only the System Manager.

3.8.3. Configure Menu

It is recommended that the System Administrator configure an HPSS system traversing the Configure menu in top down order since some configuration items have a dependency on others.

Subsystems. Opens the *Subsystems* list window where a list of all configured subsystems can be viewed, new subsystems can be configured or existing subsystems can be deleted. Additionally, from the *Subsystems* list window, accounting statistics can be viewed and reports can be generated.

Policies. This submenu lists the policy types that can be configured for HPSS.

- **Accounting.** Opens the *Accounting Policy* window allowing configuration and management of the accounting policy. Only one accounting policy is allowed.

- **Location.** Opens the *Location Policy* window allowing configuration and management of the location policy. Only one location policy is allowed.
- **Logging.** Opens the *Logging Policies* list window allowing configuration and management of the logging policies.
- **Migration.** Open the *Migration Policies* list window allowing configuration and management of the migration policies.
- **Purge.** Opens the *Purge Policies* list window allowing configuration and management of the purge policies.

Storage Space. This submenu lists the storage space configurations required for HPSS. Classes of Service contain a Hierarchy, and Hierarchies are made up of a list of Storage Classes.

- **Storage Classes.** Opens the *Configured Storage Classes* list window allowing configuration and management of storage classes.
- **Hierarchies.** Opens the *Hierarchies* list window allowing configuration and management of storage hierarchies.
- **Classes of Service.** Opens the *Class of Service* list window allowing configuration and management of the classes of service.

Servers. Opens the *Servers* list window, which will facilitate server configuration and management.

Global. Opens the *Global Configuration* window allowing the configuration and management of the HPSS global configuration record. Only one global configuration is allowed.

Devices & Drives. Opens the *Devices and Drives* list window allowing configuration and management of devices and drives for use with HPSS.

File Families. Opens the *File Families* list window allowing the configuration and management of file families.

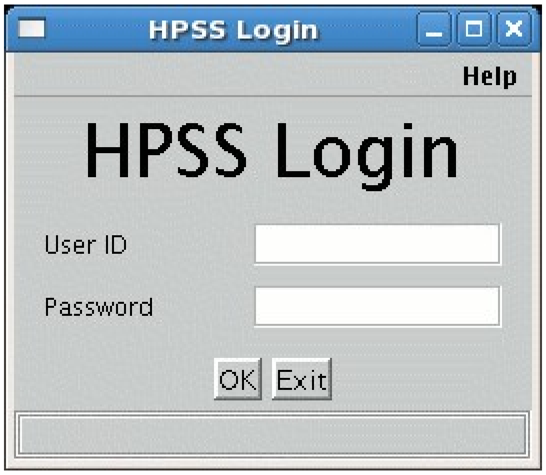
Restricted Users. Opens the *Restricted Users* list window, which will facilitate HPSS user access management.

List Preferences. This submenu contains an entry for each SSM list window. A preference record can be created or the default preference record can be modified to allow the user to customize each SSM list window's data view by using filtering methods. See Section 3.10: *SSM List Preferences* on page 69 for more information.

3.9. SSM Specific Windows

This section describes the *HPSS Login* window, the *HPSS Health and Status* window, and the SSM information windows.

3.9.1. HPSS Login



The *HPSS Login* window appears after starting the **hpssgui** script. The user must supply a valid HPSS user name and password in order to access SSM and monitor HPSS.

If a login attempt is unsuccessful, review the user session log for an indication of the problem. See the **hpssadm** or **hpssgui** man pages for more information about the user session log.

Field Descriptions

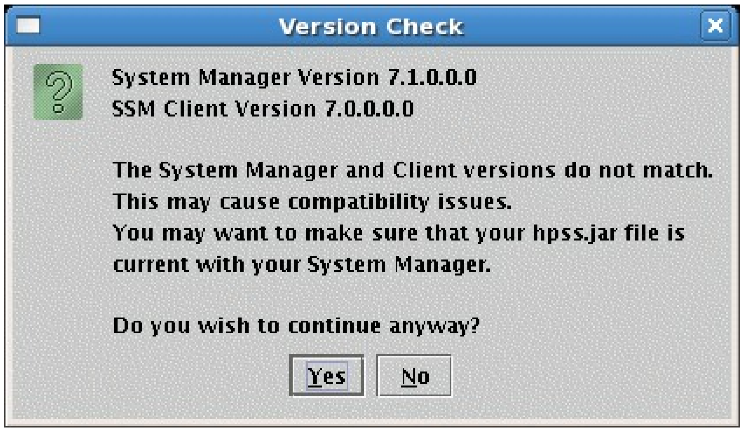
User ID. Enter a valid user ID here.

Password. Enter the password for the user ID.

OK. Attempt to contact the System Manager and authenticate the user. If the attempt is successful, the *HPSS Health and Status* widow will open. Otherwise, an error message will be displayed in the status bar.

Exit. Close the *HPSS Login* window and terminate the **hpssgui** client session.

If the System Manager and SSM Client versions do not match then the following window will be displayed after the user logs in;



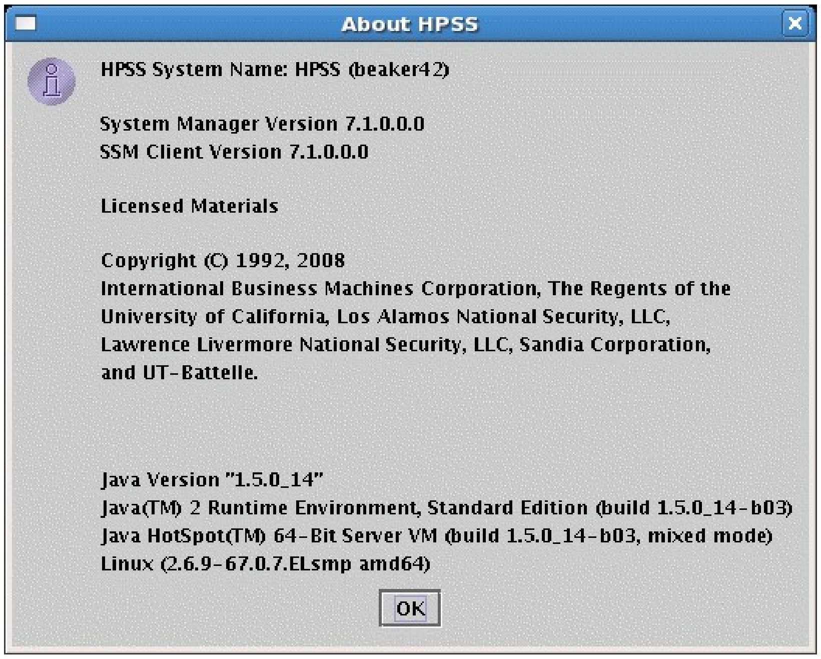
System Manager Version 7.1.0.0.0
SSM Client Version 7.0.0.0.0

The System Manager and Client versions do not match.
This may cause compatibility issues.
You may want to make sure that your hpss.jar file is current with your System Manager.

Do you wish to continue anyway?

The user may choose to continue logging in or to exit. However, as the dialog says, running with mismatched versions may cause compatibility problems.

3.9.2. About HPSS



HPSS System Name: HPSS (beaker42)

System Manager Version 7.1.0.0.0
SSM Client Version 7.1.0.0.0

Licensed Materials

Copyright (C) 1992, 2008
International Business Machines Corporation, The Regents of the University of California, Los Alamos National Security, LLC, Lawrence Livermore National Security, LLC, Sandia Corporation, and UT-Battelle.

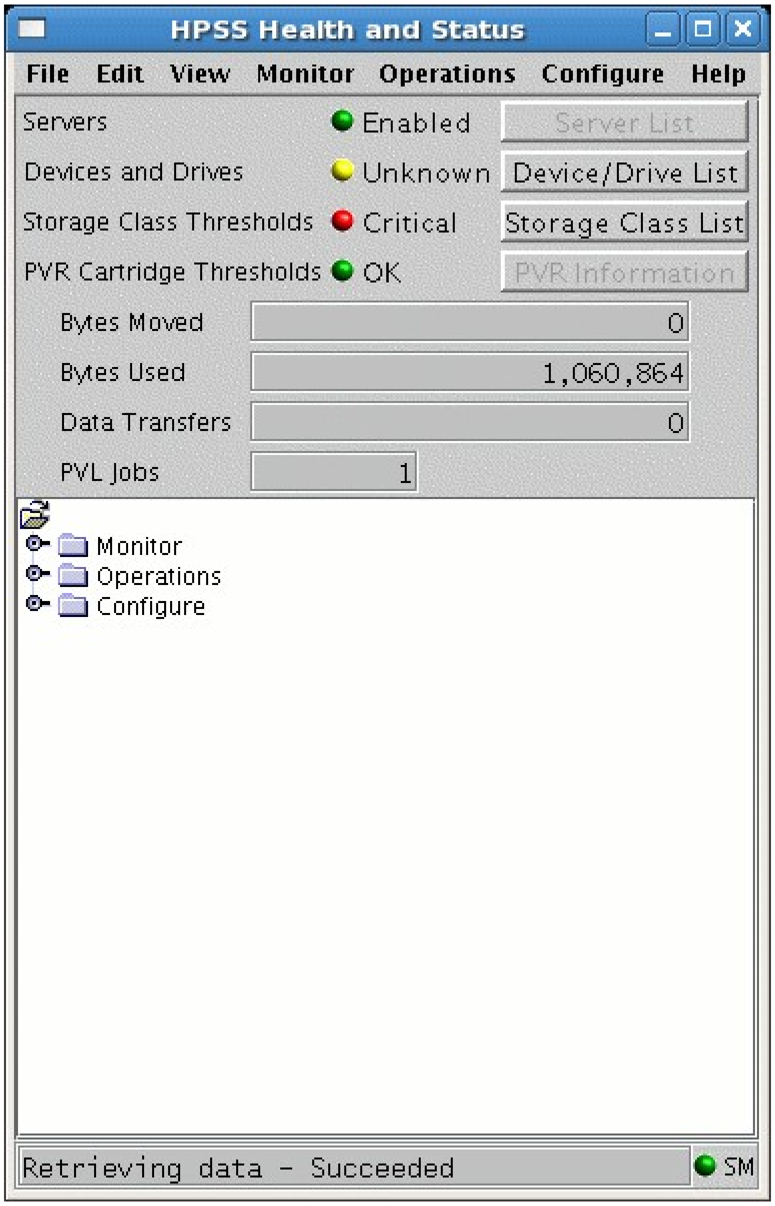
Java Version "1.5.0_14"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0_14-b03)
Java HotSpot(TM) 64-Bit Server VM (build 1.5.0_14-b03, mixed mode)
Linux (2.6.9-67.0.7.ELsmp amd64)

The *About HPSS* window displays version information and a portion of the HPSS copyright statement. The *About HPSS* window is accessible by selecting the Help menu's "About HPSS" submenu from any of the hpssgui windows.

The HPSS System Name and System Manager Version are not displayed when the *About HPSS* window is requested from the *HPSS Login* window. These two pieces of information are not available until the user actually logs into the System Manager.

Differences in the System Manager and SSM Client versions may indicate that the client and/or System Manager code should be updated.

3.9.3. HPSS Health and Status



Servers	● Enabled	Server List
Devices and Drives	● Unknown	Device/Drive List
Storage Class Thresholds	● Critical	Storage Class List
PVR Cartridge Thresholds	● OK	PVR Information
Bytes Moved	0	
Bytes Used	1,060,864	
Data Transfers	0	
PVL Jobs	1	

```
Retrieving data - Succeeded
```

When a user successfully connects to the System Manager through the Login window, the *HPSS Health and Status* window replaces the Login window on the screen. The *HPSS Health and Status* window will remain on the screen until the user exits or logs out. It provides the main menu and displays information about the overall status of HPSS .

The *HPSS Health and Status* window is composed of several high-level components, each of which is discussed in its own section below.

3.9.3.1. SM Server Connection Status Indicator

The SM connection status indicator is located in the bottom right corner of the status bar. When the SM icon is red, the client's connection to the System Manager is lost; when it is green, the connection is active.

3.9.3.2. HPSS Status

On the upper section of the *HPSS Health and Status* window are four status fields that represent the aggregate status of the HPSS system. These fields are:

Servers. Displays the most severe status reported to SSM by any HPSS server.

Devices and Drives. Displays the most severe status as reported to SSM for any configured Mover device or PVL drive.

Storage Class Thresholds. Displays the most severe status reported by the MPS for the Active Storage Class space usage. SSM assumes that all thresholds are "OK" until it receives contrary information.

PVR Cartridge Thresholds. Displays the most severe status of cartridge usage reported by any configured PVR. SSM assumes that all thresholds are "OK" until it receives contrary information.

For the **Servers** and **Devices and Drives** fields, possible status values in increasing order of severity are:

- Normal - No problem reported.
- Unknown - SSM cannot determine the true status due to communication problems or other difficulties.
- Suspect - There may or may not be a problem.
- Minor - A problem has been encountered by the server or device, but it does not significantly affect the HPSS operation.
- Major - A problem has been encountered by the server or device that may degrade HPSS operation.
- Critical - A problem has been encountered by the server or device that may disrupt HPSS operation until the problem is resolved.

For the **Storage Class Thresholds** field, the possible status values are:

- OK - No problem reported.
- Warning - A threshold has been reported as crossing its warning limit.
- Critical - A threshold has been reported as crossing its critical limit.
- Stale - Migration Purge Server is down or SSM has not received an update from the MPS.

For the **PVR Cartridge Thresholds** field, the possible status values are:

- OK - No problem reported.
- Warning - A threshold has been reported as crossing its warning limit.
- Critical - A threshold has been reported as crossing its critical limit.
- Unknown or Stale - PVR is down or SSM has not received an update from the PVR.

As problems are resolved or returned to normal, the status fields will automatically reflect the changes.

In addition to the text which describes the status, these fields are displayed with colored icons. The icon color depicts that status as follows:

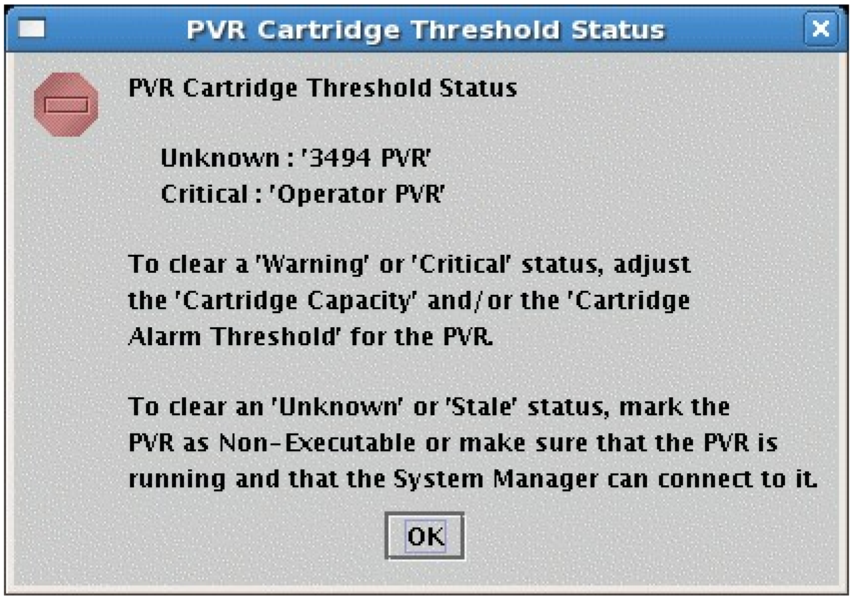
- Red - Major and Critical problems
- Magenta – Minor problems
- Yellow - Unknown, Stale, Suspect, and Warning problems
- Green - Normal, no problem

Click on the button to the right of the status icon to get more details.

For Servers, Devices and Drives, and Storage Class Thresholds the button will open the corresponding SSM list window in sick list mode. Once this window is open, use it to get detailed status information on the sick entries, assuming the HPSS Servers are still healthy enough to respond to SSM requests.

See Section 3.10: *SSM List Preferences* on page 69 for more information on the sick list mode.

For PVR Cartridge Thresholds the button will display a message dialog with information about the PVRs that have cartridge threshold issues. This message dialog will look like the following;



PVR Cartridge Threshold Status

Unknown : '3494 PVR'
Critical : 'Operator PVR'

To clear a 'Warning' or 'Critical' status, adjust the 'Cartridge Capacity' and/or the 'Cartridge Alarm Threshold' for the PVR.

To clear an 'Unknown' or 'Stale' status, mark the PVR as Non-Executable or make sure that the PVR is running and that the System Manager can connect to it.

The status section of the *HPSS Health and Status* window can be hidden from view by selecting the View menu item and unchecking the HPSS Status checkbox.

3.9.3.3. HPSS Statistics

The HPSS Statistics fields are located in the middle section of the *HPSS Health and Status* window and display the number of bytes moved, bytes used, data transfers, and current PVL jobs in the system. The number of bytes moved and number of data transfers indicate the data accumulated by the Movers since startup or data reset.

HPSS Statistics fields show general trends in HPSS operations; the numbers are not all-inclusive. Some values may fluctuate up and down as servers are started or shut down. Some values, such as **Bytes Moved**, can be reset to zero in individual Movers and by SSM users.

Bytes Moved. Total bytes moved as reported by all running Movers.

Bytes Used. Total bytes stored on all disk and tape volumes as reported by all running Core Servers.

Data Transfers. Total data transfers as reported by all running Movers.

PVL Jobs. Total jobs reported by the PVL.

The statistics section of the *HPSS Health and Status* window can be hidden from view by selecting the View menu item and unchecking the HPSS Statistics checkbox.

3.9.3.4. Menu Tree

The Menu Tree section of the *HPSS Health and Status* window displays a tree structure that mirrors the structure of the Monitor, Operations and Configure menus. The menu tree can be fully expanded so that the user can view all the menu options for monitoring, configuring or operating HPSS. Selecting a leaf of the menu tree results in the same response as selecting the same item from the *HPSS Health and Status* menu bar. The user can also choose to expand only those branches of the menu tree in which he is interested. The collapsed and expanded state of the branches on the menu tree can be toggled by clicking on the branch icons.

The Menu Tree can be hidden from view by selecting the View menu item and unchecking the Menu Tree checkbox.

3.9.3.5. File Menu

All SSM data windows have the File menu. See Section 3.6: *Common Window Elements* on page 50 for a description of the File Menu options that appear on all SSM data windows. The following File menu options are unique to the *HPSS Health and Status* window:

- **Cascade** – Select the Cascade menu option to rearrange the SSM windows, placing them on the screen starting in the upper left-hand corner and cascading downward toward the lower right-hand corner of the user's desktop. When the cascading has been completed, the *HPSS Health and Status* window will be centered on the screen and brought to the foreground.
- **Close All** - Select Close All to close all SSM windows except the *HPSS Health and Status* window. To close the *HPSS Health and Status* window, the user must select Logoff or Exit.
- **Logoff** - Select Logoff to exit the SSM session and close all SSM windows. The **hpssgui** script will continue to be active and will still use the same user's session log. The *HPSS Login* window will appear. The user can login again to reconnect to the System Manager.
- **Exit** - Select Exit to exit the SSM session and close all SSM windows. The **hpssgui** script will terminate and the user's session log will be closed. The user must rerun the **hpssgui** script to access the *HPSS Login* window and reconnect to the System Manager.

3.9.3.6. View Menu

The View Menu is only available on the *HPSS Health and Status* window. The View Menu offers the user the ability to hide or display elements of the *HPSS Health and Status* window in order to optimize the viewable area. Under the View Menu there is a menu item and checkbox for each window element that can be hidden. If the box contains a check mark then the corresponding section of the *HPSS Health and Status* window that displays this element will be visible. If the checkbox is empty, then the element is hidden from the window view. Clicking on the checkbox will toggle the visible state of the window element.

HPSS Status. Toggle the HPSS Status menu option to hide or view the HPSS Status section of the *HPSS Health and Status* window.

HPSS Statistics. Toggle the HPSS Statistics menu option to hide or view the HPSS Statistics section of the *HPSS Health and Status* window.

Menu Tree. Toggle the Menu Tree menu option to hide or view the Menu Tree section of the *HPSS Health and Status* window.

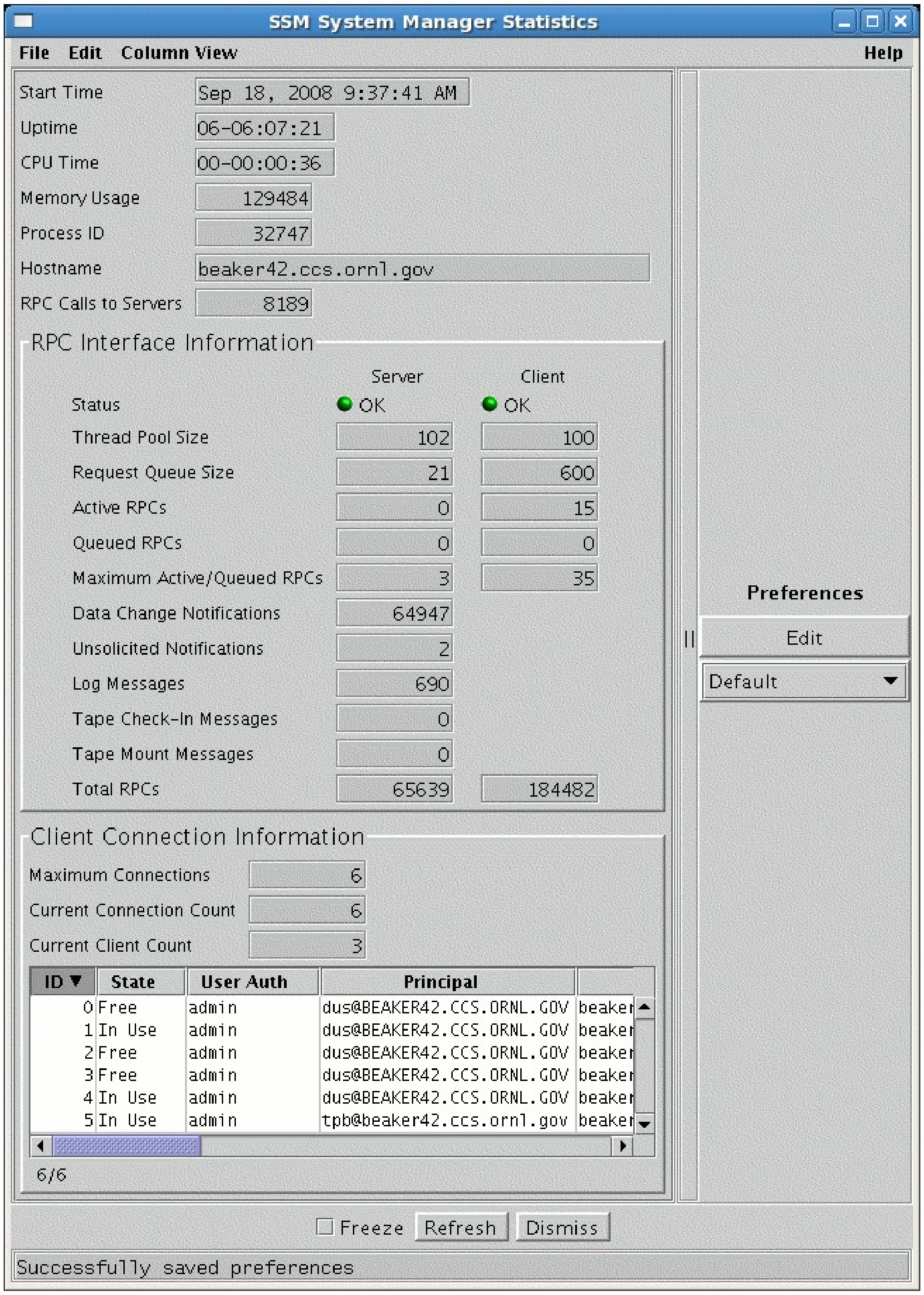
3.9.4. SSM Information Windows

These windows describe the System Manager and the user's **hpssgui** session.

3.9.4.1. System Manager Statistics Window

This window displays the statistics about the number of RPC calls, notifications and messages that the System Manager has processed since its Start Time. While the window is open, the statistics are updated at timed intervals. The information in this window is intended for use in analyzing System Manager performance or in troubleshooting. It is of limited use in everyday production situations.

To open the window, from the *HPSS Health and Status* window Monitor menu select SSM Information, and from its submenu select System Manager Statistics.



Start Time	Sep 18, 2008 9:37:41 AM
Uptime	06-06:07:21
CPU Time	00-00:00:36
Memory Usage	129484
Process ID	32747
Hostname	beaker42.ccs.ornl.gov
RPC Calls to Servers	8189

RPC Interface Information

	Server	Client
Status	● OK	● OK
Thread Pool Size	102	100
Request Queue Size	21	600
Active RPCs	0	15
Queued RPCs	0	0
Maximum Active/Queued RPCs	3	35
Data Change Notifications	64947	
Unsolicited Notifications	2	
Log Messages	690	
Tape Check-In Messages	0	
Tape Mount Messages	0	
Total RPCs	65639	184482

Client Connection Information

Maximum Connections	6
Current Connection Count	6
Current Client Count	3

ID ▼	State	User Auth	Principal	
0	Free	admin	dus@BEAKER42.CCS.ORNL.GOV	beaker
1	In Use	admin	dus@BEAKER42.CCS.ORNL.GOV	beaker
2	Free	admin	dus@BEAKER42.CCS.ORNL.GOV	beaker
3	Free	admin	dus@BEAKER42.CCS.ORNL.GOV	beaker
4	In Use	admin	dus@BEAKER42.CCS.ORNL.GOV	beaker
5	In Use	admin	tpb@beaker42.ccs.ornl.gov	beaker

6/6

Successfully saved preferences

Field Descriptions

Start Time. The time the System Manager was started.

Uptime. The elapsed clock time since the System Manager was started.

CPU Time. The amount of CPU time that the System Manager has consumed.

Memory Usage. The amount of memory that the System Manager is currently occupying.

Process ID. The process id of the System Manager.

Hostname. The name of the host where the System Manager is running.

RPC Calls to Servers. The number of RPCs the System Manager has made to other HPSS servers.

RPC Interface Information. Information about the server and client RPC interfaces. The server interface is used by other HPSS servers to contact the System Manager. The client interface is used by the **hpssgui** and **hpssadm** programs to contact the System Manager. There are 2 columns of data, one for the server interface and one for the client interface. Not all fields are available for both interfaces. The fields include:

- **Status.** Current status of the thread pool and request queue of the RPC interface. The Status can be:
    - **OK** – The number of Active RPCs is less than the **Thread Pool Size**. There are enough threads in the thread pool to handle all current RPCs with spare ones left over.
    - **Warning** – The number of Active RPCs is greater than the **Thread Pool Size**. The number of Queued RPCs is less than 90% of the Request Queue Size. There aren't enough threads to handle all the current RPCs and some are having to wait in the queue, but the queue is big enough to hold all the waiters.
    - **Critical** – The number of Queued RPCs is greater than or equal to 90% of the **Request Queue Size**. There aren't enough threads to handle all the current RPCs, some are having to wait in the queue, and the queue is getting dangerously close to overflowing, at which point any new RPCs will be rejected.

- **Thread Pool Size.** The maximum number of RPCs that can be active at any one time. For the server RPC interface this value is determined by the **HPSS_SM_SRV_TPOOL_SIZE** environment variable. For the client RPC interface this value is determined by the **Thread Pool Size** field defined on the *Core Server Configuration* window. Refer to Section 5.1.1.2: *Error: Reference source not found* on page 92.

- **Request Queue Size.** The maximum number of RPC requests that can be queued and waiting to become active. For the server RPC interface this value is determined by the **HPSS_SM_SRV_QUEUE_SIZE** environment variable. For the client RPC interface this value is determined by the **Request Queue Size** field on the *Core Server Configuration* window. Refer to Section 5.1.1.2: *Error: Reference source not found* on page 92.

- **Active RPCs.** The number of RPCs that are currently active. To be active an RPC must have been assigned to a thread in the thread pool.

- **Queued RPCs.** The number of RPCs that are waiting in the request queue to be assigned to a thread in the thread pool.

- **Maximum Active/Queued RPCs**. The maximum number of RPC requests that were active (in the thread pool) or queued (in the request queue) at the same time. This value can be used to help tune the **Thread Pool Size** and **Request Queue Size** for the RPC interface. If the **Maximum Active/Queued RPCs** is greater than the **Thread Pool Size** you might consider increasing the

Thread Pool Size and/or **Request Queue Size** to help with the System Manager performance. However, increasing these 2 parameters could cause the System Manager to require more memory.

- **Data Change Notifications.** The number of data change notifications received from servers. (Server RPC Interface only).
- **Unsolicited Notifications.** The number of notifications which the System Manager received from other HPSS servers but which it did not request from them. (Server RPC Interface only).
- **Log Messages**. The number of log message notifications processed. (Server RPC Interface only).
- **Tape Check-In Messages.** The number of tape check-in request notifications received. (Server RPC Interface only).
- **Tape Mount Messages**. The number of tape mount request notifications received. (Server RPC Interface only).
- **Total RPCs.** Total number of RPCs processed by the RPC interface.

Client Connection Information. Information about clients that have connected to the System Manager.

- **Maximum Connections.** The maximum number of client connections that the System Manager was handling at any one time. Each client can have multiple connections. The default connections per client is 2. Each client can specify the number of connections using the -Dhpss.ssm.SMConnections Java command line option.
- **Current Connection Count.** The number of client connections currently being handled by the System Manager.
- **Current Client Count.** The number of clients currently connected to the System Manager. This will be the number of entries in the Client List (below) with an In Use state.
- **Client List.** The list of clients connected to the System Manager. The entries that appear in the client list will be for (up to) the last 64 (SSM_CLIENT_MAX) clients that have connected to the System Manager.
  - **ID.** Slot identifier.
  - **State.** Clients displayed in the client list can have one of two states: **In Use** and **Free**.
    - **In Use.** The clients that are currently connected to the System Manager.
    - **Free.** The clients that were once connected but are no longer connected (active).

Once the client list contains SSM_CLIENT_MAX **In Use** and/or **Free** entries, then the oldest **Free** slot (the client that has been disconnected the longest) is given to the next client. Once all the slots are **In Use**, then the client table is full; no new clients can connect until one of the **In Use** slots becomes **Free** after a client disconnects.

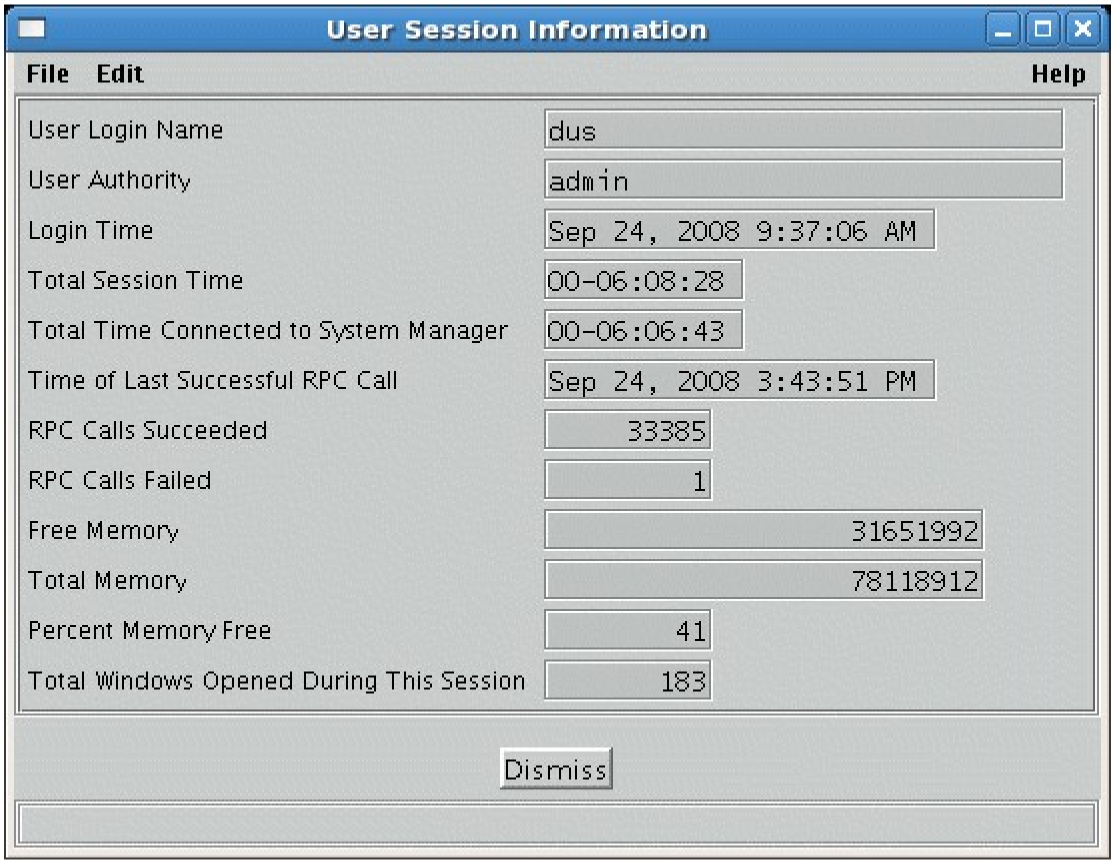
  - **User Auth.** The client user's authorization level: **admin** or **operator**. See Section 3.3: *Configuration and Startup of hpssgui and hpssadm* on page 34 for more details.
  - **Principal.** The user's principal name.

- **Hostname.** The name of the host where the client is running.
- **Connections.** The number of RPC connections this client has to the System Manager.
- **Start Time.** The time that the client connected to the System Manager.
- **Connect Time.** The elapsed time since the client connected to the System Manager.
- **Idle Time.** The elapsed time since the System Manager received an RPC from the client.
- **Cred Refreshes.** The number of times the principal's credentials have been refreshed since the client has been connected to the System Manager.
- **RPC Calls.** The number of RPCs that the client has made to the System Manager.
- **RPC Waits.** The number of RPCs that the client has currently “waiting” in the System Manager. These are the RPCs which are active and connected to the System Manager but which have not yet completed.
- **Client UUID.** The UUID for the client.

3.9.4.2. User Session Information Window

The *User Session Information* window displays memory and network statistics for the current SSM user session. The window is updated in timed intervals as data changes.

To open the window, from the *HPSS Health and Status* window Monitor menu select SSM Information, and from its submenu select User Session Information.



User Login Name	dus
User Authority	admin
Login Time	Sep 24, 2008 9:37:06 AM
Total Session Time	00-06:08:28
Total Time Connected to System Manager	00-06:06:43
Time of Last Successful RPC Call	Sep 24, 2008 3:43:51 PM
RPC Calls Succeeded	33385
RPC Calls Failed	1
Free Memory	31651992
Total Memory	78118912
Percent Memory Free	41
Total Windows Opened During This Session	183

Field Descriptions

User Login Name. The name that the user entered as the User ID when logging into HPSS.

User Authority. The authority level that the user has in order to perform operations in SSM. This can be admin or operator.

Login Time. The time that the user logged into SSM.

Total Session Time. The amount of time that the user has had the graphical user interface running.

Total Time Connected to System Manager. The amount of time that the user has been connected to the System Manager. This is the time that has elapsed since the login was successful.

Time of Last Successful RPC Call. The time that the last RPC call was made to the System Manager which completed successfully.

RPC Calls Succeeded. The number of successful RPC calls made by this user session.

RPC Calls Failed. The number of unsuccessful RPC calls made by this user session.

Free Memory. Amount of free memory available in the **hpssgui** program.

Total Memory. Amount of total memory available in the **hpssgui** program.

Percent Memory Free. The ratio of free memory to total memory in the **hpssgui** process.

Total Windows Opened During This Session. The number of windows created during the current user session.

3.10. SSM List Preferences

When a user logs into SSM, the System Manager reads the saved preferences file and loads them into the SSM Session, if they exist.

Each SSM list type has a Default preferences record. The Default preferences configuration is set so that the more commonly used columns are displayed. The following lists have a preferences record:

- Alarms & Events
- Classes of Service
- Devices & Drives
- File Families
- Filesets & Junctions
- Hierarchies
- Logging Policies
- Migration Policies
- Purge Policies
- PVL Jobs
- Restricted Users
- RTM Summary
- Servers
- SM Clients
- Storage Classes, Active
- Storage Classes, Configured
- Subsystems
- Tape Check-Ins
- Tape Mounts

The *Servers*, *Devices and Drives*, and *Active Storage Classes* list windows also have Sick preferences which display all items with abnormal status.

Default preferences can be modified but cannot be deleted. Sick preferences may not be modified or deleted.

Preferences are saved on the client node in the directory specified by the “configuration path” **hpssgui** command line argument ("-i"). This option can also be set using the environment variable

HPSSGUI_USER_CFG_PATH or the configuration file entry HPSS_SSM_USER_PREF_PATH. If this option is not specified, the default value is **<client node>:<user.home>/hpss-ssm-prefs**. The user must have permissions to create the preferences file in the directory.

Preferences windows contain filters for controlling the data displayed in the list window. Columns in the list window can be rearranged and resized by dragging columns and column borders on the list window itself. Data in the list can be sorted ascending or descending based on any column in the list by clicking in that column's header. Such changes are kept up to date in the preferences file transparent to the user, without the need to click on a **Save** button.

Columns can be hidden or redisplayed through the Column View menu on each list window.

The *Tape Mount Requests* and *Tape Check-In Requests* windows have Auto Popup checkboxes. The states of these checkboxes, as well as those in the View Menu of the *HPSS Health and Status* window are stored in the preferences file transparent to the user.

If a user's preferences file becomes obsolete, the current version of the user interface software will convert the old preferences to the current format.

Checkbox Filters

Checkbox filters apply to columns that have a limited set of display values. The checkbox filters are grouped by column name and contain a checkbox for each allowed value. If the checkbox is selected then all rows containing the value will be displayed on the corresponding SSM list window. At least one checkbox filter must be selected in each group.

Text Filters

Columns that can be filtered by their text content are listed in the Text Filters section of the preference window. Users can control which rows are displayed by entering a Java regular expression into one or more of the **Text Filter** fields. If the **Text Filter** field is blank then no filtering on the field will occur.

The preference window will verify that the text entered is a valid Java regular expression before allowing the user to save the preferences.

For example, if the list contains storage class names as follows

```
SClass_4-way_Disk
SClass_4-way_Tape
Sclass_4-way_Archive
Sclass_Disk
```

entering a Java regular expression of ".*4-way.*" would result in the "Sclass_Disk" entry to be filtered out of the display.

For a complete description on how to use regular expressions in Java, please refer to the following web page; http://java.sun.com/j2se/1.5.0/docs/api/java/util/regex/Pattern.html.

Button Options

Save as ... Save the current preference settings to a new named preference. A popup window will be displayed so that the new name can be entered.

Save. Save the current preference settings to the preferences file using the same preference name.

Delete. Delete the currently displayed preference settings from the preferences file. Sick and Default preference settings cannot be deleted.

Reload. Reread the preference settings from the preferences file.

Apply. Apply the current preference configuration to the parent SSM list window. Pressing **Apply** does not save the changes to the preferences file. To save the preference settings, press the **Save** or **Save as ...** button. The **Apply** button is only available when the Preferences window is accessed via its parent SSM List window (i.e., it is not available when the Preferences window is accessed from the Configure menu).

Show List. Brings the window to which these preferences apply to the foreground. The **Show List** button is only available when the Preferences window is accessed via its parent SSM List window.

Dismiss. Close the window.

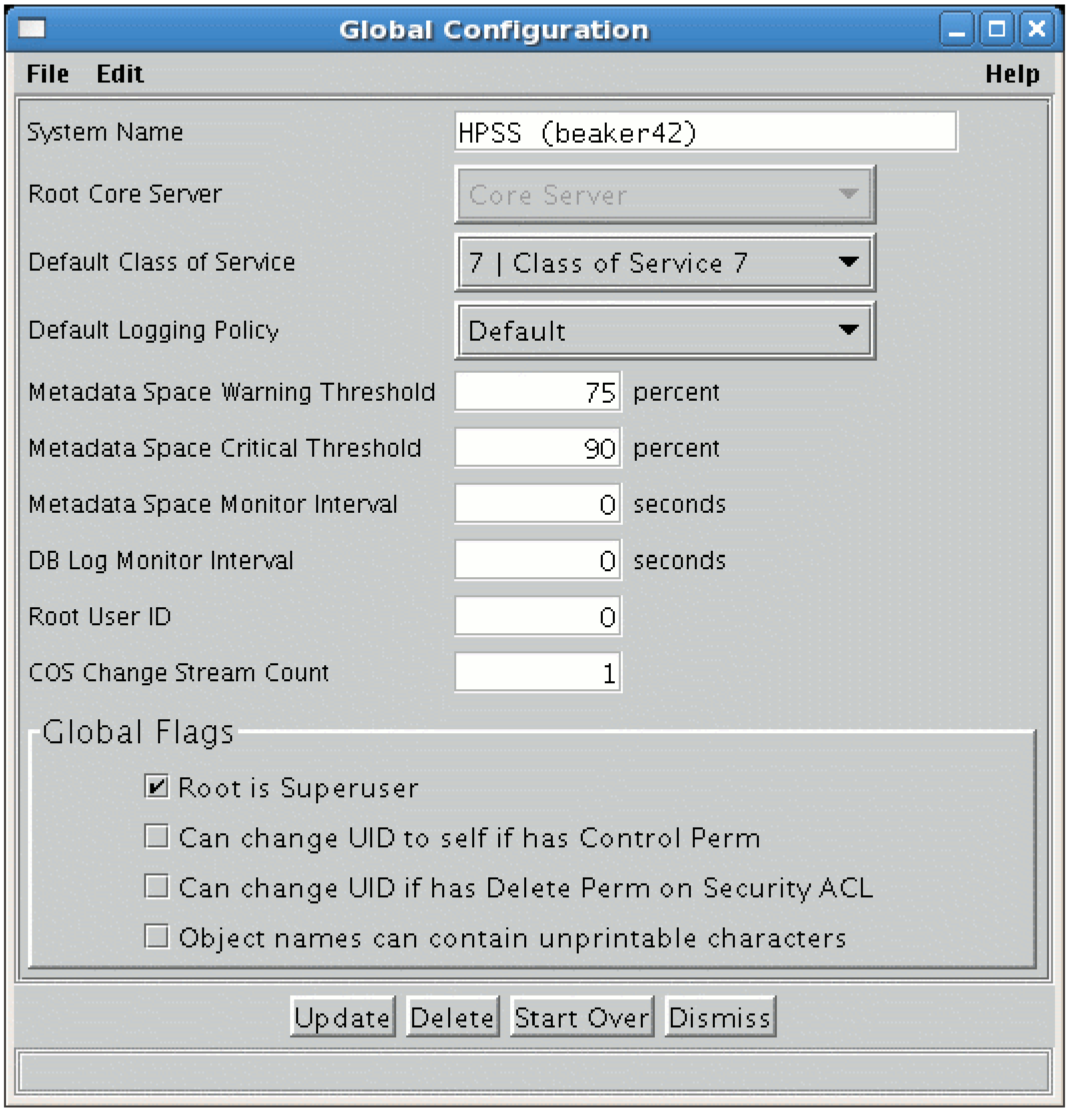
To modify an existing preference configuration, select the Preference Name from the drop down list and press the **Reload** button. Modify the configuration then press the **Save** button. To create a new Preference configuration, open an existing configuration, make the modifications, press the **Save as ...** button then enter a unique value for the new preference name and press the **OK** button.

Chapter 4. Global & Subsystem Configuration

This chapter discusses two levels of system configuration: global and storage subsystem. The global configuration applies to the entire HPSS installation while subsystem configurations apply only to servers and resources allocated to storage subsystems.

For new HPSS systems, it is recommended that the first step of configuration be the partial definition of subsystems. The subsystems should be defined at this point except for their Gatekeeper and Default COS information. The Global Configuration should be created after Server configuration is complete. Last of all, the subsystem configuration is completed by adding the definitions of the Gatekeeper and Default COS. See Section 1.3: *HPSS Configuration Roadmap (New HPSS Sites)* on page 1.3.

4.1. Global Configuration Window



System Name	HPSS (beaker42)
Root Core Server	Core Server
Default Class of Service	7 | Class of Service 7
Default Logging Policy	Default
Metadata Space Warning Threshold	75 percent
Metadata Space Critical Threshold	90 percent
Metadata Space Monitor Interval	0 seconds
DB Log Monitor Interval	0 seconds
Root User ID	0
COS Change Stream Count	1

Global Flags

- [x] Root is Superuser
- [ ] Can change UID to self if has Control Perm
- [ ] Can change UID if has Delete Perm on Security ACL
- [ ] Object names can contain unprintable characters

This window allows you to manage the HPSS global configuration record. Only one such record is permitted per HPSS installation. To open the window, on the *Health and Status* window select the Configure menu, and from there the Global menu item.

Field Descriptions

System Name. An ASCII text string representing the name for this HPSS system.

Root Core Server. The name of the Core Server which manages the root fileset ("/") in the HPSS namespace.

Advice - The root Core Server should be selected with care. Once it is chosen it cannot be changed as long as the chosen server exists. If the root Core Server must be changed, the current root Core Server will have to be deleted before SSM will allow another root Core Server to be selected.

Default Class of Service. The name of the default Class of Service (COS). Core Servers will store new files in this COS when the HPSS client does not specify a COS or provide hints on the creation request. This default can be overridden by each storage subsystem.

Advice - If the COS chosen for the Default Class of Service is deleted, be sure to change the Default Class of Service before deleting the COS.

If disk is used as the Default COS, it should be defined so that a large range of files sizes can be handled For this reason it may be reasonable to set the default COS to a tape COS.

Default Logging Policy. The name of the default logging policy. This policy will be used for all servers that do not have a specific log policy configured.

Advice - Care should be taken to choose a descriptive name (such as "DEFAULT LOG POLICY") which is unlikely to conflict with the name of a real HPSS server. Once this new policy is created, it may be selected as the Default Log Policy.

Alternately, you may skip this step until after one or more server specific log policies are created. Then a Default Log Policy can be chosen from one of the server specific log policies.

Metadata Space Warning Threshold. Provides a default value for the metadata warning threshold. When the space used in any DB2 tablespace exceeds this percentage, the Core Server will issue a periodic warning message. This value can be overridden for each storage class on a storage subsystem basis (see Section 4.2.1: *Subsystems List Window* on page 74 for more information).

Metadata Space Critical Threshold. Provides a default value for the metadata critical threshold. When the space used in any DB2 tablespace exceeds this percentage, the Core Server will issue a periodic critical alarm. This value can be overridden for each storage class on a storage subsystem basis (see Section 4.2.1: *Subsystems List Window* on page 74 for more information).

Metadata Space Monitor Interval. The Core Server will check metadata usage statistics at the indicated interval, specified in seconds. The minimum value for this field is 1800 seconds (30 minutes). A value of 0 will turn off metadata space monitoring. This field may be overridden on the Storage Subsystem configuration.

DB Log Monitor Interval. The Core Server will check consistency of Database Logs and Backup Logs at the indicated interval, specified in seconds. The logs are consistent if both primary and backup log directories exist and contain log files with the same names. The minimum value for this field is 300 seconds (5 minutes). A value of 0 will turn off DB Log monitoring. This field may be overridden on the

Storage Subsystem configuration.

Root User ID. The UID of the user who has root access privileges to the HPSS namespace. This only applies if the **Root Is Superuser** flag is set.

COS Change Stream Count. The number of background threads that run in the Core Server to process Class of Service change requests. This field may be overridden on the Storage Subsystem configuration.

Global Flags:

Root Is Superuser. If checked, root privileges are enabled for the UID specified in the **Root User ID** field. If the box is not checked, the UID specified in the **Root User ID** field will not have root privileges. Root access privileges grant the specified user the same access rights to each namespace object as the object's owner.

Can change UID to self if has Control Perm. If checked, then when a user has control access to an object, the user can take over ownership of the object by changing the UID of the object to the UID associated with the user.

Can change UID if has Delete Perm on Security ACL. If checked, then when a user is listed with delete permission in the security ACL of the CORE server that owns the object, the user can change the UID of the object to any valid UID.

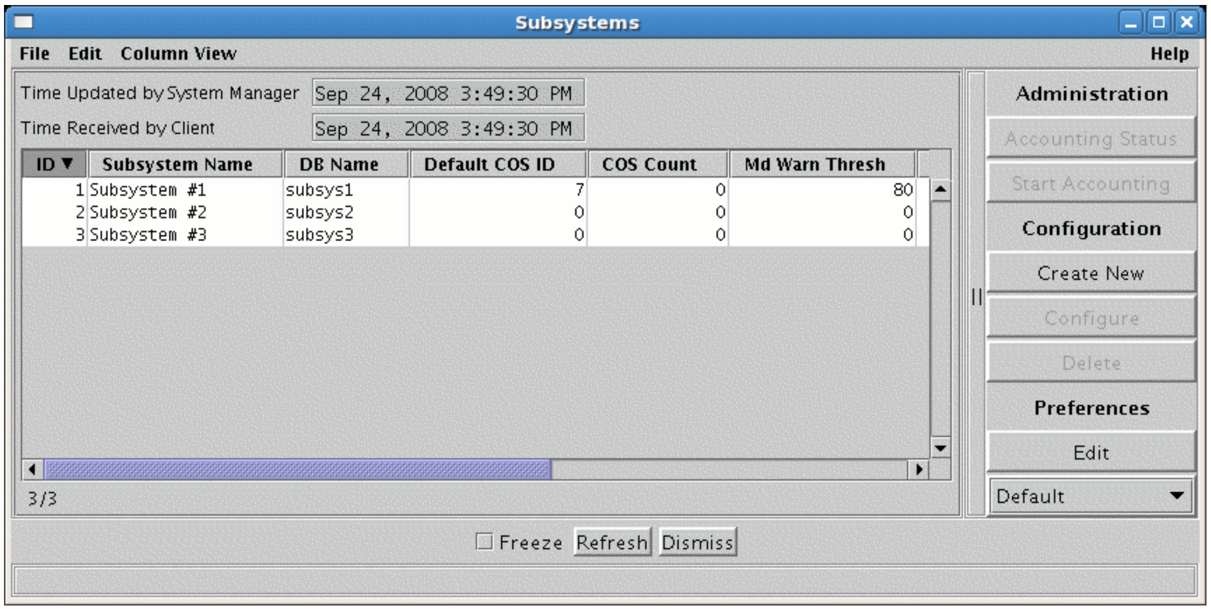
Object names can contain unprintable characters. If checked, then users of the HPSS system may create objects (e.g., files, directories, etc.) with names containing unprintable characters as viewed through the 7 bit ASCII character set. If this option is off, any attempt to name an object using unprintable characters will be disallowed. The range of printable characters is 0x20 (blank) through 0x7E (tilde) in the 7 bit ASCII character set.

4.2. Storage Subsystems

Every HPSS system consists of at least one subsystem. This chapter provides instructions and supporting information for creating, modifying, and deleting them.

For a conceptual overview of subsystems, refer to the *HPSS Installation Guide,* specifically *Storage Subsystems* in Sections 2.2.7 and 2.3.3.

4.2.1. Subsystems List Window



Time Updated by System Manager: Sep 24, 2008 3:49:30 PM

Time Received by Client: Sep 24, 2008 3:49:30 PM

ID	Subsystem Name	DB Name	Default COS ID	COS Count	Md Warn Thresh
1	Subsystem #1	subsys1	7	0	80
2	Subsystem #2	subsys2	0	0	0
3	Subsystem #3	subsys3	0	0	0

3/3

This window lists all the subsystems in the HPSS system and provides the ability to manage these subsystems. To open the window, from the *Health and Status* window select the Configure menu, and from there select the Subsystems menu item.

To create a new subsystem, click on the Create New button. To configure an existing subsystem, select it from the list and click on the Configure button. When creating or configuring a subsystem, the *Storage Subsystem Configuration* window will appear.

To delete an existing subsystem, select it from the list and click on the Delete button.

Field Descriptions

Subsystem List table columns. For details on the columns listed for each subsystem, refer to Section 4.2.3: *Storage Subsystem Configuration Window* on page 76.

Administration Buttons

Accounting Status - Opens the Accounting Status window which displays the status and statistics from the last accounting run. See Section 13.2.2.1: *Generating an Accounting Report* on page 332 for more information. You must first select a configured subsystem by clicking on the subsystem entry from the list of displayed subsystems.

Start Accounting - Start the Accounting utility to generate an accounting report. See the Accounting Status window for updates from the utility. You must first select a subsystem by clicking on the subsystem entry from the list of displayed subsystems.

Configuration Buttons

Create New - Create a new storage subsystem definition by opening a *Storage Subsystem* window containing default values for a new subsystem.

Configure - Open the selected subsystem configuration for editing. This button is disabled unless a subsystem is selected in the list.

Delete - Delete the selected subsystem(s). This button is disabled unless a subsystem is selected in the list.

Always contact HPSS customer support before deleting a subsystem definition. An improperly deleted subsystem can cause serious problems for an HPSS system. Refer to Section *4.2.5 Deleting a Storage Subsystem on* Page *81.*

Related Information

HPSS Installation Guide, Section 2.2.7: *Storage Subsystems* and Section 2.3.3: *HPSS Storage Subsystems*

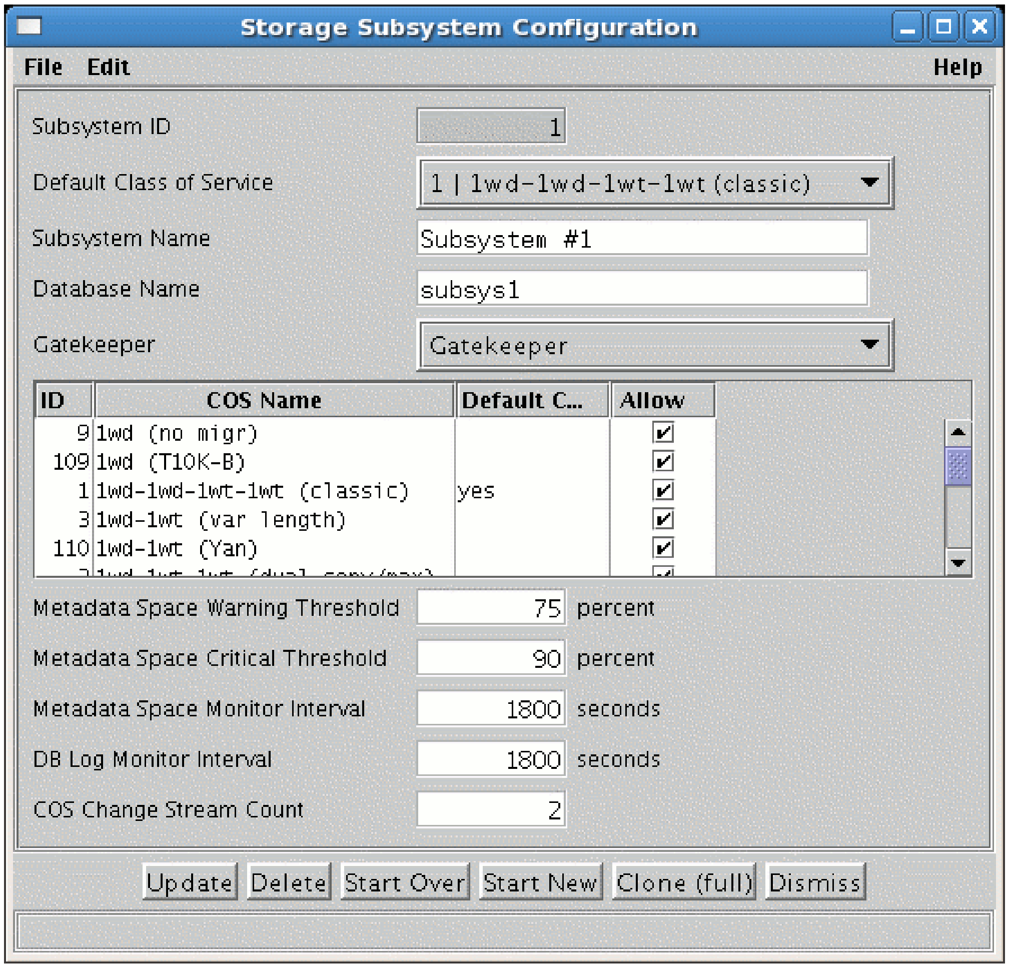
Section 13.2.2.1: *Generating an Accounting Report* on page 332

4.2.2. Creating a New Storage Subsystem

The steps detailed in the following sections should be followed to create a new storage subsystem within an HPSS system. These steps are:

1. Create the Storage Subsystem Metadata (Section 4.2.3.1: *Create Storage Subsystem Metadata* on page 79)
2. Create the Storage Subsystem Configuration (Section 4.2.3.2: *Create Storage Subsystem Configuration* on page 79)
3. Create the Storage Subsystem Servers (Section 4.2.3.3: *Create Storage Subsystem Servers* on page 80)
4. Assign a Gatekeeper to the subsystem, if required (Section 4.2.3.4: *Assign a Gatekeeper if Required* on page 80)
5. Assign Storage Resources to the Storage Subsystem (Section 4.2.3.5: *Assign Storage Resources to the Storage Subsystem* on page 80)
6. Create the Storage Subsystem Fileset and Junction (Section 4.2.3.6: *Create Storage Subsystem Fileset and Junction* on page 80)
7. Configure Migration and Purge Policy Overrides (Section 4.2.3.7: *Migration and Purge Policy Overrides* on page 81)
8. Configure Storage Class Threshold Overrides (Section 4.2.3.8: *Storage Class Threshold Overrides* on page 81)

4.2.3. Storage Subsystem Configuration Window



Subsystem ID	1
Default Class of Service	1 | 1wd-1wd-1wt-1wt (classic)
Subsystem Name	Subsystem #1
Database Name	subsys1
Gatekeeper	Gatekeeper

ID	COS Name	Default C...	Allow
9	1wd (no migr)		✓
109	1wd (T10K-B)		✓
1	1wd-1wd-1wt-1wt (classic)	yes	✓
3	1wd-1wt (var length)		✓
110	1wd-1wt (Yan)		✓

Metadata Space Warning Threshold	75	percent
Metadata Space Critical Threshold	90	percent
Metadata Space Monitor Interval	1800	seconds
DB Log Monitor Interval	1800	seconds
COS Change Stream Count	2	

This window allows an administrator to manage the configuration of a storage subsystem.

The Add button is only displayed during the creation of a new configuration. The Update button is displayed when an existing configuration is being modified.

To open this window for creation of a new subsystem, click the Create New button on the *Subsystems* window. To open this window for an existing subsystem, select the subsystem from the *Subsystems* window and click the Configure button.

Field Descriptions

Subsystem ID. A unique, positive integer ID for the storage subsystem. This field may only be set at create time. The default value is the last configured subsystem ID number plus 1. The default subsystem ID can be overwritten but if the new number already exists, an attempt to save the configuration will fail.

Default Class of Service. This value overrides the Global Default Class of Service specified on the *Global Configuration* window. This default Class of Service applies to this storage subsystem only. The default value is "None".

Advice - HPSS Core Servers use a default Class of Service to store newly-created files when the HPSS *user does not specify a COS or any hints with the creation request. The global configuration specifies a default COS for an entire HPSS installation. Selecting a COS on the storage subsystem configuration window allows the global value to be overridden for a particular subsystem.*

If the field is blank, the global default COS will be used. If no Classes of Service are configured, this value can be updated after the Classes of Service are in place.

Subsystem Name. The descriptive name of the storage subsystem. This field may be set only when the storage subsystem is created. The name should be unique and informative. It can contain a character string up to 31 bytes long. The default value is "Subsystem #<ID>".

Database Name. The name of the database to be used to store the metadata for the storage subsystem.

Gatekeeper. The default value is "None".

Advice - If an appropriate Gatekeeper has not yet been configured, simply leave this configuration entry blank. It can be updated after the Gatekeeper is in place.

Allowed COS list. A list of Classes of Service that can be used by this subsystem. To allow a COS to be used by this subsystem, the corresponding checkbox must be selected in the Allow column of the list. At least one COS must always be selected. The user will not be permitted to de-select the COS defined to be the Default Class of Service. If this subsystem configuration does not have a Default Class of Service defined, then the COS chosen as the Global Configuration's Default Class of Service cannot be de-selected.

Note that a newly created COS will not appear in the selection list until the Core Server and Migration Purge Server associated with the subsystem have been recycled. When new Classes of Service are added, the initial allowed state for that COS is determined by the current setting for the other Classes of Service. If all previous Classes of Service were allowed, the new COS will be allowed. Otherwise, the new COS will be disallowed.

Advice - By default, the servers in a subsystem are able to use any configured COS. This table allows an administrator to prevent a subsystem from using particular Classes of Service.

When a new Class of Service is added to a system, it will automatically be enabled for all subsystems which have no disabled Classes of Service. It will be disabled in all other subsystems. If this is not the desired configuration, the COS will have to be allowed/disallowed for each subsystem individually.

Disallowing all Classes of Service in a subsystem is not permitted.

Metadata Space Warning Threshold. The Core Server in this subsystem will issue a warning alarm periodically and set its Opstate to Major when the percentage of used space in any DB2 tablespace in this subsystem's database exceeds this value.

Metadata Space Critical Threshold. The Core Server in this subsystem will issue a critical alarm periodically and set its Opstate to Critical when the percentage of used space in any DB2 tablespace in this subsystem's database exceeds this value.

Metadata Space Monitor Interval. The Core Server for this subsystem will check the metadata usage statistics at the indicated interval, specified in seconds. If a value of 0 is specified, the Global Configuration setting will be used for this storage subsystem. The minimum value for this field is 1800 seconds (30 minutes).

DB Log Monitor Interval. The Core Server will check consistency of Database Logs and Backup Logs at the indicated interval, specified in seconds. The logs are consistent if both primary and backup log directories exist and contain log files with the same names. The minimum value for this field is 300 seconds (5 minutes). A value of 0 will turn off DB Log monitoring. This field may be overridden on the Storage Subsystem configuration.

COS Change Stream Count. The number of background threads that run in the Core Server to process Class of Service change requests. If a value of 0 is specified, the Global Configuration setting will be used for this storage subsystem.

4.2.3.1. Create Storage Subsystem Metadata

Before creating the subsystem metadata, you must have created the subsystem database that will be used by the subsystem and have created appropriate tablespaces for the database.

You should review and perform the steps in the following sections in the *HPSS Installation Guide* before allocating metadata space for the new subsystem:

- 2.3.3 *HPSS Storage Subsystems*
- 3.5.2 *HPSS Infrastructure Storage Space*
- 3.5.3 *HPSS Filesystems*
- 3.5.4 *HPSS Metadata Space*
- 5.3.2 *Install and Configure HPSS – Secondary Subsystem Machine*

4.2.3.2. Create Storage Subsystem Configuration

Use the following steps to create a storage subsystem configuration:

1. Decide on a unique descriptive name for the new storage subsystem. SSM will automatically choose the name "Subsystem #N", where N is the subsystem ID selected by SSM. The name of the new storage subsystem may be changed by the administrator at subsystem configuration time only.
2. From the *Health and Status* window, select Configure/Subsystems from the menu. You will then be presented a window that lists currently configured subsystems. Select the Create New button to bring up another window to configure the new subsystem.
3. Enter the name you have chosen for the subsystem in the Subsystem Name field and enter the ID of the subsystem in the subsystem ID field. Enter the name of the database you have chosen to contain the tables for this subsystem in the Database Name field.
4. Decide which Classes of Service are to be supported by the new storage subsystem. SSM will automatically select all Classes of Service to be supported, but the administrator can modify these choices at any time. Also decide on a default Class of Service for this storage subsystem. By default SSM leaves this field blank, which means that the default Class of Service specified in the global configuration will apply to the new storage subsystem as well. The administrator may choose to override the global value by using the subsystem configuration value at any time.
5. Decide whether gatekeeping or account validation are needed for this storage subsystem. If either is required, a Gatekeeper will need to be configured for this storage subsystem. If the required

Gatekeeper is already configured, simply add it to your storage subsystem's configuration. However, if it is not yet configured, it will be necessary to wait until Section 4.2.3.4: *Assign a Gatekeeper if Required* on page 80 to add the Gatekeeper.

6. Set the metadata space thresholds and the update interval. Typical values are 75 for warning, 90 for critical and 300 to have the metadata space usage checked every 300 seconds.
7. Set the DB Log Monitor Interval. Minimum value is 300 seconds, typical value is 1800 seconds.
8. Press the Add button to store the configuration.

4.2.3.3. Create Storage Subsystem Servers

The new storage subsystem must contain a single Core Server. If migration and purge services will be needed then a Migration Purge Server is also required. See the following sections for instructions on configuring these new servers:

- Section 5.1: *Server Configuration* on page 87
- Section 5.1.1: *Core Server Specific Configuration* on page 96
- Section 5.1.2: *Migration/Purge Server (MPS) Specific Configuration* on page 101

On each server's basic configuration window, be sure to assign the server to the new storage subsystem.

4.2.3.4. Assign a Gatekeeper if Required

If gatekeeping or account validation are needed and an appropriate Gatekeeper is not already configured, a new Gatekeeper will be required. See the *HPSS Installation Guide,* Section 3.7.3: *Gatekeeper*, Section 5.1.2: *Gatekeeper Specific Configuration* on page 98 for instructions on configuring the Gatekeeper.

Be sure to assign this Gatekeeper to the appropriate storage subsystem by choosing it from the Gatekeeper selection list on the appropriate Storage Subsystem Configuration window.

4.2.3.5. Assign Storage Resources to the Storage Subsystem

Storage resources are tied to a storage subsystem through storage classes. To determine which storage classes belong to a particular subsystem, look up the configuration for each Class of Service available to the storage subsystem. Each Class of Service is tied to one storage hierarchy which is in turn tied to some number of storage classes. Storage resources must then be assigned to the storage classes which are used in the referenced hierarchies. See Section 8.1.2: *Creating Storage Resources* on page 234 for details.

4.2.3.6. Create Storage Subsystem Fileset and Junction

The HPSS namespace consists of filesets that are connected by junctions. The top (root directory) of the HPSS namespace is the RootOfRoots fileset managed by the Root Core Server. The rest of the namespace is built by pasting filesets together using junctions.

Since each subsystem has its own Core Server, it also has its own root fileset. This fileset needs to be connected to the HPSS namespace in order for files to be stored in the new subsystem. This is accomplished by creating a junction from the HPSS namespace to the root fileset of the new subsystem.

See Chapter 10: Filesets and Junctions for more information.

4.2.3.7. Migration and Purge Policy Overrides

The migration and purge policies contain two elements, the basic policy and the storage subsystem specific policies. This can be seen on the *Migration Policy* and *Purge Policy* windows. If a given migration or purge policy does not contain any subsystem specific policies, then the basic policy applies across all storage subsystems and no other configuration is needed. If it is desired for migration or purge to behave differently than specified in the basic policy in a given storage subsystem, then a storage subsystem specific policy should be created for that subsystem. A subsystem specific policy allows some or all of the values in the basic policy to be overridden for the given storage subsystem. See Section 6.4.2: *Migration Policy Configuration* on page 182 and Section 6.5.2: *Purge Policy Configuration* on page 190 for more information.

4.2.3.8. Storage Class Threshold Overrides

The warning and critical thresholds given on the *Storage Class Configuration* window apply across all storage subsystems unless specified otherwise. The Subsystem Thresholds button on the *Configured Storage Class* list window allows the default thresholds to be overridden for specified storage subsystems. See Section 6.1.1: *Configured Storage Classes Window* on page 157 for more information.

4.2.4. Modifying a Storage Subsystem

If modifications are made to an existing Storage Subsystem configuration, the Core Server and the Migration Purge Server for the subsystem must be recycled.

4.2.5. Deleting a Storage Subsystem

Always contact HPSS customer support before deleting a subsystem definition. An improperly deleted subsystem can cause serious problems for an HPSS system.

It is critical that no files or directories exist within a storage subsystem before it is deleted. It is important to verify that all of the DB2 metadata tables associated with the storage subsystem being deleted are empty.

To verify that all files have been removed from the subsystem, perform the following steps:

1. Run the **dump_acct_sum** utility on the subsystem. Be sure to specify the subsystem ID with the -s option as this utility defaults to subsystem 1. The output from the utility should indicate that 0 files exist in the subsystem.
2. As a second verification do the following:
   A. Run the db2 command line utility program.
   B. Connect to the subsystem database (e.g., **connect to subsys1**).
   C. Set the schema (e.g., **`set schema hpss`**).
   D. Issue the following SQL command:

   ```
   db2> select count(*) from bitfile
   ```

   The result of the command should indicate 0 rows in this table.

E. Issue the following SQL command:

```
db2> select count(*) from nsobject
```

The result of the command should indicate 2 rows in this table.

3. If any of these checks gives an unexpected result, do not delete the subsystem. Contact HPSS customer support.

When deleting an existing storage subsystem, it is critical that all of the different configuration metadata entries described in section 4.2.3: Storage Subsystem Configuration Window on page 76 for the storage subsystem be deleted. If this is not done, configuration metadata entries associated with the subsystem will become orphaned when the subsystem configuration itself is deleted. This situation is difficult to correct after the subsystem is deleted.

Once it has been verified that the storage subsystem contains no files or directories, deleting the subsystem may be accomplished by reversing the steps used to create the subsystem. These steps are listed in the order which they should be performed below. For information on each step, see the corresponding instructions under Section 4.2.2: *Creating a New Storage Subsystem* on page 76.

1. Delete all storage resources assigned to the subsystem's Core Server. See Section 8.2.1: *Deleting Storage Resources* on page 240.
2. Delete all filesets and junctions used to link the storage subsystem into the HPSS name space. See Section 10.6: *Deleting a Junction* on page 316.
3. Delete all storage class subsystem-specific thresholds associated with the storage subsystem being deleted. This is done by setting these thresholds back to the storage class “default” thresholds.
4. Delete all subsystem-specific migration and purge policy entries associated with the storage subsystem being deleted. See Section 6.4.2.4: *Deleting a Migration Policy* on page 188 and Section 6.5.4: *Deleting a Purge Policy* on page 193 for more information.
5. Delete all Core Servers, Migration Purge Server and DMAP Gateway Server residing within the storage subsystem being deleted. See Section 5.1.1: *Deleting a Server Configuration* on page 123.
6. Delete the subsystem configuration.
7. Remove the database associated with the subsystem.

Chapter 5. HPSS Servers

Most HPSS Server administration is performed from the SSM graphical user interface *Servers* list window. Each HPSS server has an entry in this list.

5.1. Server List

This window facilitates management of the configured HPSS servers. From this window, an HPSS server can be started, shut down, halted, reinitialized, and notified of repair. Once a server is up and running, SSM monitors and reports the server state and status. Information on running servers may also be viewed and updated via this window.

If multiple entries are selected in the server list when a server operation is invoked, the operation will be applied to all selected servers (although some servers don't support all operations).

The server entries displayed on the window can be sorted by each column category. To sort the server entries by status, for example, click on the Status column title. The actual display can vary greatly by the setting of window preferences. The Column View menu item can be used to select which columns of the table are visible, and Preferences can be used to further refine and filter which servers are visible. Preferences can also be saved and automatically reloaded by SSM when new sessions are started (see Section 3.10: *SSM List Preferences* on page 69 for more information).

At times, the server list may update quite frequently with new Status or Opstate information. If you select either of these columns for sorting, servers are likely to change their positions in the list rapidly. Sometimes this makes the list hard to use, in which case you should consider selecting a more static column for sorting or check the **Freeze** button to keep the list from updating.

Field Descriptions

Server List.

This is the main portion of the window which displays various information about each server.

ID. A unique numerical value assigned by SSM when each server starts. This value can change each time SSM is restarted. In the default preferences, this column is not shown.

Status. The server execution and connection status as determined by SSM. The reported status will be one of the following:

- Connected - Server is up and running and communicating normally with SSM.
- Up/Unconnected - Server is up and running (according to the Startup Daemon) but SSM cannot connect to it. Server cannot be completely controlled and monitored through SSM.
- Down - Server is down. SSM can be used to start the server.
- Indeterminate - The server's state cannot be determined by SSM and the Startup Daemon is either not running or not connected to SSM.
- Check Config - SSM detected an incomplete or inconsistent configuration for the server.

The server's configuration should be carefully reviewed to ensure that it is correct and complete. Check the *Alarms and Events* window and the HPSS log file to view SSM alarm messages related to configuration problems. This situation can be caused by:

- A DB2 record required by the server is missing or inaccessible.
- The principal name configured for the server does not match the HPSS_PRINCIPAL_* environment variable for the server's type.

- Not Executable - The server is configured as non-executable. (Note: the server could still be running. It may have been started outside the control of SSM, or it may have been running when its executable flag was changed.) SSM will not monitor the server's status.
- Deleted - The server configuration has been deleted. (Note: deleted servers will be removed from the list when SSM is restarted.) In the default preferences, this Status type is filtered off.

In addition to the above status values, the **Status** field also reports the transient status for the server as the result of the user request on the server as follows:

- Starting... - The server is being started.
- Stopping... - The server is being shut down gracefully.
- Halting... - The server is being forcibly halted.
- Reiniting... - The server is reinitializing.
- Connecting... - SSM is trying to establish a connection to the server.
- Repairing... - The server is repairing its states and statuses.

A server that is configured to execute and is running should have a Connected status. If its status is anything other than Connected (excluding the transient status values), one of the following actions should be taken:

- If the server status is Up/Unconnected - Monitor the server status closely for a few minutes. SSM will periodically try to establish a connection with the server. The **Force Connect** button can be used to speed this process.
- If the server status is **Down** - Use the **Start** button to start the server. The Startup Daemon will ensure that only one instance of the server is running.
- If the server status is Indeterminate - Verify whether the server is running using an operating system command such as "ps". If the server is not running, start it. If the server is running, ensure that the Startup Daemon configured for the same node is running and has a connection to SSM. If the Startup Daemon is not running, start it using the **/opt/hpss/bin/rc.hpss** script on the appropriate node. Otherwise, use the **Force Connect** button to establish the connections for the server and the Startup Daemon. If this does not correct the server's status, review the *Alarms and Events* window to search for problems that the server and SSM may have reported. In addition, review the HPSS logs for the server's and SSM's log messages to help determine the problems.

SSM will periodically poll the servers' execution and connection status and update the **Status** fields when there are any changes. The rate of these updates will depend on the client's refresh rate (see the hpssgui/hpssadm man pages for more details).

If a server is configured as executable but is not running, SSM will treat it as an error. Therefore, if a server is not intended to run for an extended period, its Executable flag should be unchecked. SSM will stop monitoring the server and will not report the server-not-running condition as a critical error. This will also help reduce the work load for SSM.

Type. Indicates the type of the server in acronym form. Possible values include CORE, GK, LOGC, LOGD, LS, MPS, MOVER, PVL, various PVR values, SSMSM, and SUD. See the glossary for the meanings of these acronyms.

Subtype. Indicates the subtype of the server, sometimes in acronym form. Not all servers have subtypes. Possible values (particularly PVR subtypes) include SCSI, STK, 3494, Operator, AML, and 3584 LTO. See the glossary for the meanings of these acronyms.

Subsystem. The name of the storage subsystem to which the server is assigned. Only servers of type Core and MPS should have a subsystem name. For all other server types this column should be blank.

Op State. The operational state of the server as reported by the server itself:

- Enabled - The server is operating normally.
- Disabled - The server is not operational, usually due to a shutdown/halt request.
- Suspect - The server may have a problem.
- Minor - The server encountered a problem that does not seriously affect the HPSS operation.
- Major - The server encountered a problem that may seriously impact the overall HPSS operation.
- Broken - The server encountered a critical error and shut itself down.
- Unknown- The server's state is unknown to SSM. SSM should be able to obtain an Operational State for the server but cannot because SSM cannot communicate with it, and the server has not set its state to Disabled or Broken. The server may or may not be running. The reason may simply be that the server is marked executable but is not running.
- Invalid - SSM has obtained an Operational State from the server, but its value is not recognized as a valid one. This may indicate a bug in the server or a communications problem which is garbling data.
- None - The server does not have an Operational State (because the server is not executable, for example), or its Operational State is, by design, unobtainable.

SSM will periodically poll the server's operational state and update the **Op State** field in the *Servers* window when there are any changes. The rate of the polling is controlled by the SSM client list refresh rate (see the hpssgui/hpssadm man pages).

Server Name. The descriptive name of the server.

Host. The name of the host on which the server is running.

Execute Host. The **Execute Hostname** field from the server's basic configuration record. This field is intended to specify the hostname on which the server is supposed to run; however, no checking is done to verify if the server is actually running on the specified host. This field is only used by the SSM to locate the Startup Daemon that manages this server. The field displayed must match exactly the **Execute Hostname** field specified in the configuration record of the startup daemon that is to manage this server.

UUID. The universal unique identifier for the server. In the default preferences, this column is not shown.

Administration Buttons.

This group of buttons allows you to perform administrative operations on the selected servers. All of the buttons are disabled unless one or more servers are selected and the operation is applicable to at least one of the selected servers.

After pressing one of these buttons, you will usually be prompted to confirm your request before it is carried out. If you make an obviously invalid request, such as asking to start a server which is not executable, SSM will tell you about it, but otherwise no harm is done. You could, for example, select a range of servers and ask to start them all, knowing that some of the selected servers are already running. Those that are not running will be started, but nothing will be done to those that are already running.

The status bar at the bottom of the window displays a message when the requested operation has begun. For information on the success or failure of the operation, monitor the Status and Op State columns for the server.

Start – Start the selected servers. The System Manager will notify the Startup Daemon to start the selected servers.

Reinitialize – Send a “reinitialize” command to the selected servers. Note that not all HPSS servers support reinitialization. If a server does not support reinitialization, the button will not be sensitive.

Mark Repaired – Clear displayed error status in the selected servers. Sometimes server states such as the Operational State will continue to indicate an error condition after the cause of the error has been fixed. If so, you can use the **Mark Repaired** button to clear its error states. Note that this does not do anything in hardware or software to actually repair an error; it just asks the server to clear its error condition. If you mark a server repaired when it still has a problem, the error states will be cleared but may quickly return.

Shutdown – Command the selected servers to shutdown. This command should be the first you use to shutdown servers. After issuing this command, wait a minute or two for the server to complete the shutdown process. Some servers shutdown very quickly after receiving the command, others, particularly the Core Server, may require two minutes or more to shutdown. During this time the server is attempting to finish work it has started, while rejecting new work. Be patient; watch the *Alarm and Events* window for messages indicating the server has terminated.

Force Halt – Command the selected servers to stop immediately. This should be done only if a shutdown request has failed to stop the servers, or if the intention is to shut servers down as quickly as possible. This request will cause a SIGKILL signal to be sent to the selected servers if all other shutdown methods have failed. This command is meant to be used as a last resort if a server hangs up or otherwise won't respond to the Shutdown command.

Force Connect - Request the System Manager to immediately attempt to connect to the selected servers. The System Manager routinely attempts to connect to any unconnected servers; using this button will simply cause the next attempt to occur right away, instead of after the normal retry delay.

Information Buttons.

These buttons allow you to open information windows for servers. They will be sensitive only when the selected server supports a particular information window.

Since the requested information is obtained by calling the selected server, the server must normally have a Connected status for the request to succeed.

Basic Info - Opens the *Basic Server Information* window for the selected server.

Specific Info - Opens the type-specific server information window for the selected server.

Configuration Buttons.

These buttons allow you to start server configuration tasks.

Create New - Allows an administrator to configure a new server by selecting a server type and then filling in a new server configuration. This control is actually not a button, but a pulldown list, allowing the administrator to select the type of new server to be created. It is always enabled.

Configure - Opens the configuration window(s) for the selected server(s).

Delete - Deletes the selected server(s) from the system. A confirmation dialog will appear to confirm the action. Before deleting a server, see the warnings and considerations in Section 5.1.1: *Deleting a Server Configuration* on page 123.

5.1. Server Configuration

The following HPSS servers should be configured. A Gatekeeper server is required only if the site wishes to do gatekeeping or account validation. See the *HPSS Installation Guide*, section 2.3.2: *HPSS Servers* for a description of the purpose of each server:

- Core Server
- Migration/Purge Server
- Gatekeeper (optional)
- Location Server
- Log Client
- Log Daemon
- Physical Volume Library
- Physical Volume Repository
- Mover

- Storage System Manager
- Startup Daemon (on each host where an HPSS server will be executing)

The fields of the *Server Configuration* window are divided into the following sections. The Basic Controls section is at the top of the window and the other sections are on individual tabs:

- **Basic Controls. S**erver identification and type information.
- **Execution Controls.** Information required to properly control the server's execution.
- **Interface Controls.** Information required for communication between the server and other HPSS servers and clients.
- **Security Controls.** Security settings.
- **Audit Policy.** Object event audit settings.
- **Log Policy**. Information for controlling the types of log messages generated by the server.
- **Specific.** Server type-specific settings (only for some server types).

The server specific configuration section contains configuration parameters specific to a particular server type. Not every type of server has a specific configuration section. The following types of servers have a server specific configuration section in the *Server Configuration* window:

- Core Server
- Gatekeeper
- Log Client
- Log Daemon
- Migration/Purge Server
- Mover
- Physical Volume Repository

Although the Location Server does not have a server specific configuration section in the Server Configuration window, there are additional configuration steps outside SSM necessary whenever a Location Server is added or modified.

Details about the specific configuration section and the additional configuration required for each of these server types are described in:

- Section 5.1.1: *Core Server Specific Configuration* on page 96
- Section 5.1.2: *Gatekeeper Specific Configuration* on page 98
- Section 5.1.3: *Location Server Additional Configuration* on page 99
- Section 5.1.4: *Log Client Specific Configuration* on page 100
- Section 5.1.1: *Log Daemon Specific Configuration* on page 101
- Section 5.1.2: *Migration/Purge Server (MPS) Specific Configuration* on page 101

- Section 5.1.3: *Mover Specific Configuration* on page 102
- Section 5.1.1: *Physical Volume Repository (PVR) Specific Configuration* on page 109
- Details about all of the other sections on this window, which apply to all server types, are described in Section 5.1.1: *Common Server Configuration*.

To view the *Server Configuration* window for an existing server, bring up the *Servers* list window and select the desired server. Then click the **Configure** button to bring up the configuration window for that server.

To modify a server configuration, bring up the server's configuration window, modify the desired fields, and click the **Update** button. Some configuration fields may be set by the administrator only at server creation time. These fields will be editable/settable when creating new servers, but will become display-only fields (not editable or settable) when looking at existing configurations. For the modified configuration data to be recognized by the server, the server must be reinitialized (if supported) or restarted.

There are two ways to create a new server:

1. To create a new server using standard defaults as a starting point, open the *Servers* list window, click **Create New** and select a server type. This opens a *Server Configuration* window with default values for the selected server type. Make the desired customizations to the window and click the **Add** button.
2. To create a new server that has attributes similar to those of an existing server, select the existing server, click **Configure** and then select **Clone (partial)**. This opens a *Server Configuration* window with some default values copied from the existing server. Make the desired customizations to the window and click the **Add** button.

There are two ways to delete a server. Before deleting a server, see the warnings and considerations in Section 5.1.1: *Deleting a Server Configuration* on page 123. Then, to delete the server:

1. From the *Servers* list window, select the server and click the **Delete** button, or
2. Bring up the *Server Configuration* window and click the **Delete** button.

In both cases you will be prompted to confirm the deletion of the server.

5.1.1. Common Server Configuration

These sections of the *Server Configuration* window are common to all servers.

5.1.1.1. Basic Controls

The Basic Controls section of the *Server Configuration* window is common to all servers. In the example window above, the server displayed is a Core Server.

Field Descriptions

Server Name. A unique descriptive name given to the server. Ensure that the Server Name is unique. A server's descriptive name should be meaningful to local site administrators and operators, in contrast to the server's corresponding UUID, which has meaning for HPSS. For HPSS systems with multiple subsystems it is very helpful to append the subsystem ID to the Server Name of subsystem-specific servers. For instance, "Core Server 1" for the Core Server in subsystem 1.

Before modifying this field, check whether this server is using the default log policy or its own custom log policy (see Section 5.1.1.1: *Log Policy* on page 95). If the server is using its own custom log policy, modify it to use the default log policy. Skipping this step will cause an extraneous log policy to remain in the system.

Server ID. A UUID that identifies a server to HPSS. A unique value is automatically generated and displayed for new servers. While you are permitted to modify this field by hand, such a practice is not recommended because of the danger that you could type in a UUID which is not unique within the entire HPSS realm. This field may only be modified in Add mode.

Server Type. The type of the HPSS Server.

Server Subtype. The subtype of the selected server. This field is only used by the PVR servers to specify the type of PVR (e.g. STK).

Storage Subsystem. Name of the HPSS Storage Subsystem to which this server is assigned. This field is required for the Core Server and Migration/Purge Server. For all other servers this field is not displayed. The following rules apply to this field:

- The Storage Subsystem Configuration must exist before configuring Core Servers or Migration/Purge Servers.
- Core Servers and Migration/Purge Servers must have a valid storage subsystem set in this field before SSM will allow the server configuration to be created.
- No more than one Core Server and one Migration/Purge Server can be assigned to any one subsystem.
- This field can only be modified in Add mode.

5.1.1.1. Execution Controls

The Execution Controls section of the *Server Configuration* window is common to all servers. In the example window above, the server displayed is a Core Server.

Field Descriptions

Execute Pathname. The UNIX file system path name to a server's executable image file. This file must reside on the node specified by **Execute Hostname.** Use the full UNIX path name; otherwise, the Startup Daemon will try to start the file out of the current working directory of the Startup Daemon.

Execute Hostname. This is the hostname of the node on which the server will execute. It must match the **Execute Hostname** of the Startup Daemon that is to manage this server. For most servers, setting this field is straightforward, but for remote Movers, this indicates the node on which the Mover administrative interface process runs (not the node where the remote Mover process runs). Note that if the **Execute Hostname** is changed, it is likely that the RPC program number will change as well. If the server affected is the System Manager, the SSM configuration file, ssm.conf, must be regenerated or edited.

Program Number. The RPC program number. This value must be unique within the node in which the server runs. The administrator cannot override the default program number value when creating or modifying a server configuration.

Version Number. The RPC version number. The administrator can not override the default version number value when creating or modifying a server configuration.

UNIX Username. The UNIX user name under which the server will run. The name must be registered in the local UNIX authentication database (e.g., **/etc/passwd**) or the HPSS local password file. The Startup Daemon will use this user ID when starting the server. If there is no such user, the Startup Daemon will not be able to start the server.

Minimum Database Connections. This field is no longer used and will be deleted in a future version of HPSS. Database connections are managed dynamically and the number of database connections a server uses is based on the activity of the server. Some servers use the database only when they start and never again. Database connections used by such servers are eventually closed so that the server uses no connections at all. If the server should need a connection at some later time, one will be created. Creating database connections is a fairly low overhead activity, while maintaining unused connections is a relatively high overhead activity, so the system tries to keep unused connections at a minimum.

Maximum Database Connections. The meaning of this field has changed in HPSS 7.1. Servers are allocated database connections dynamically, using as many as are needed to perform the work at hand. Servers will create additional database connections up to **Maximum Database Connections** without delay, on demand. Once this number of connections exist, requests for additional connections will be delayed for a short period of time to give the server a chance to reuse an existing connection. If an existing connection does not become available within this time limit, a new connection will be created and processing will resume. The server's internal value for **Maximum Database Connections** is adjusted up to this new value to reflect the larger dynamic workload.

As described above, if the workload subsides, unused database connections will eventually be closed and the server will run with the number of connections that is appropriate to the workload.

Since this setting is actually dynamic, it may be removed in a future version of HPSS.

Auto Restart Count. Maximum number of automatic restarts allowed per hour. See Section 5.2.2.4: *Automatic Server Restart* on page 151 for more details.

Executable. A flag that indicates whether an HPSS server can be executed by SSM.

Auto Startup. A flag that indicates whether SSM should attempt to start the server upon startup of SSM. The Startup Daemon must be started before SSM for this option to work.

5.1.1.2. Interface Controls

The Interface Controls section of the *Server Configuration* window is common to all servers. In the example window above, the server displayed is a Core Server.

Field Descriptions

Maximum Connections. The maximum number of clients that this server can service at one time. This value should be set based on the anticipated number of concurrent clients. Too large a value may slow down the system. Too small a value will mean that some clients are not able to connect.

Thread Pool Size. The number of threads this server spawns to handle client requests. If necessary, the default values can be changed when defining servers. Too large a value may consume server memory for no purpose. Too small a value could mean that some client connections don't get serviced in a timely manner. The Thread Pool Size should be equal to or larger than the value used for Maximum Connections.

Request Queue Size. The maximum number of requests that can be queued waiting for request threads. If the workload increases so that this value is exceeded, requests will be rejected by the server rather than queued for processing. A value of zero means to use the default queue size of 20.

Note: See Section 3.1.2: *Tuning the System Manager RPC Thread Pool and Request Queue Sizes* on page 31 for information on tuning the RPC thread pool and request queue sizes for the System Manager.

Interfaces. The information required for the server to communicate with other HPSS servers and clients over the HPSS RPC network interface. Each server type is configured with one or more interfaces. With the exception of the Authentication Mechanisms, the administrator cannot override the default values for each interface when creating or modifying a server configuration. Each interface consists of the following fields:

> **Interface Name.** The descriptive name of the interface.
>
> **Interface Id.** The UUID that identifies this interface.
>
> **Interface Version.** The interface version number.
>
> **Authentication Mechanisms.** The authentication mechanisms from which the interface will accept credentials. One or both of the mechanisms can be checked (at least one should be checked or that interface will be unusable). Each interface supports the following authentication mechanisms:
>
> - KRB5 - indicates that the interface will support Kerberos 5 authentication
> - UNIX - indicates that the interface will support UNIX authentication

5.1.1.1. Security Controls

The **Security Controls** fields define the settings for authenticated communication between the server and other HPSS servers and clients.

The Security Controls section of the *Server Configuration* window is common to all servers. In the example window above, the server displayed is a Core Server.

Field Descriptions

Principal Name. The name of the principal the server will use to authenticate.

Protection Level. The level of protection that will be provided for communication with peer applications. The higher the level of protection, the more encryption and overhead required in communications with peers. The levels, from lowest to highest, are as follows:

- Connect - Performs authentication only when the client establishes a connection with the server.
- Packet - Ensures that all data received is from the expected client.
- Packet Integrity - Verifies that none of the data transferred between client and server has been modified.
- Packet Privacy - Verifies that none of the data transferred between client and server has been modified and also encrypts the data transferred between client and server.

Authentication Service Configuration. Each server can support up to two Authentication Services. The following fields are used to define each authentication service configured for a server.

> **Mechanism.** The authentication mechanism to use when passing identity information in communications to HPSS components.
>
> - KRB5 - indicates that the server will useKerberos 5 authentication.
> - UNIX - indicates that the server will use UNIX authentication.
> - Not Configured - indicates that an authentication service has not been configured for this slot. At least one of the authentication service slots must be configured.
>
> **Authenticator Type.** The type of authenticator specified in the **Authenticator** field. The types are:
>
> - Not Configured – indicates that an authenticator has not been configured for this slot. If a mechanism is specified, an authenticator type must also be specified.
> - None – indicates no authenticator is supplied for this mechanism. This is appropriate for UNIX authentication if no keytab is used. The server's credentials will be its current UNIX identity.
> - Keytab - indicates that the authenticator is the path to a keytab file. For Kerberos authentication this is a keytab file created with Kerberos utilities. For UNIX authentication this is a keytab file created with the **hpss_unix_keytab** utility. See its man page for details. Each server can have its own keytab file, or all the servers can share a single keytab file. It is recommended that one keytab file be used for all of the servers on any given host.

Authenticator. The argument passed to the authentication mechanism indicated by the **Authenticator Type** configuration variable and used to validate communications. If it is a keytab, the server must have read access to the keytab file. Other access permissions should not be set on this file or security can be breached. For the Not Configured or None values of the **Authenticator Type**, this field can be left blank.

5.1.1.1. Audit Policy

HPSS security provides a log for certain security related events such as a change in an object's permissions, a change in an object's owner, a change of an object's name, a failed attempt to access an object, and several others (see *Field Descriptions* below for a complete list). A server's Audit Policy controls which of these events trigger log messages. It can be configured from the Audit Policy tab of the *Server Configuration* window. It is possible to request that only failures or both successes and failures be logged.

For each audit event type, selecting Failure causes only failures to be logged, selecting Total logs every audit event of that type, and selecting neither causes nothing to be logged. Selecting both has the same effect as selecting Total.

Currently the only server which generates security object event records other than AUTH is the Core Server.

For more information on auditing, see the *HPSS Installation Guide,* Section 3.9.4.5: *Security Audit*.

The Audit Policy section of the *Server Configuration* window is common to all servers. In the example window above, the server displayed is a Core Server.

Field Descriptions

- **AUTH.** Authentication events.
- **CHMOD.** Core Server object permission events.
- **CHOWN.** Core Server object owner events.
- **CREATE.** Core Server creation events.
- **LINK.** Core Server hard link creation events.
- **MKDIR.** Core Server directory creation events.
- **OPEN.** Core Server bitfile open events.
- **RENAME.** Core Server object rename events.
- **RMDIR.** Core Server directory remove events.
- **UNLINK.** Core Server object delete events.

- **UTIME.** Core Server bitfile time modified events.
- **ACL_SET.** Core Server access control list modification events.
- **CHBFID.** Core Server change bitfile identifier events.
- **BFSETATTRS.** Core Server set bitfile attribute events.

5.1.1.1. Log Policy

The server Log Policy may also be accessed from the *Logging Policies* window.

It is not necessary to define a log policy for every server. If no server-specific log policy is defined for the server, the server will use the System Default Logging policy. In this case, the Log Policy tab on the *Server Configuration* window will display the values from the System Default Logging Policy.

Note that in order for modifications to the log policy to take effect, the appropriate server must be reinitialized. In most cases, the server which must be reinitialized is the Log Client which executes on the same host as the server whose log policy was changed. The only exception is for Movers; when the log policy of a Mover is modified, that Mover itself must be reinitialized in order for the log policy changes to take effect.

See Section 9.2: *Log Policies* on page 295 for a description of the *Logging Policies* window, for detailed definitions of each log message type, and for information on the System Default Logging Policy.

The Log Policy section of the *Server Configuration* window is common to all servers. In the example window above, the server displayed is a Core Server.

Field Descriptions

***Record Types to Log*.** These log message types will be sent to the log subsystem by this server.

- **ALARM.** If selected, Alarm messages generated by the server are sent to the log. It is strongly recommended that this always be ON.
- **EVENT.** If selected, Event messages generated by the server are sent to the log. It is recommended that this always be ON.
- **REQUEST.** If selected, Request messages generated by the server are sent to the log. Request messages can easily flood the log on a busy system, so in many cases, this log type should be OFF.
- **SECURITY.** If selected, Security messages generated by the server are sent to the log. Security log messages are usually relatively few in number, so this log type should be ON.
- **ACCOUNTING.** If selected, Accounting messages generated by the server are sent to the log.
- **DEBUG.** If selected, Debug messages generated by the server are sent to the log. It is recommended that this be ON, particularly in the Core Server and Mover. Debug messages are only generated when an error occurs and they can be very helpful in determining the cause of the error.

- **TRACE.** If selected, Trace messages generated by the server are sent to the log. It is recommended that this be OFF for all servers except the Mover. These messages give detailed information about program flow and are generally of interest only to the server developer. In normal operation, logging Trace messages can flood the log with very low level information. In particular, it is important to avoid TRACE for the SSM System Manager Log Policy.
- **STATUS.** If selected, Status messages generated by the server are sent to the log.

Record Types for SSM. These log message types will be sent to the *Alarms and Events* window.

- **SSM ALARM.** If selected, Alarm messages generated by the server are sent to SSM. It is strongly recommended that this always be ON.
- **SSM EVENT.** If selected, Event messages generated by the server are sent to SSM. It is recommended that this always be ON.
- **SSM STATUS.** If selected, Status messages generated by the server are sent to SSM. It is recommended that this always be ON.

Associated Button Descriptions

Use Default Log Policy. This button is used to configure the server to use the System Default Logging Policy. The System Default Logging Policy is defined on the *Global Configuration Window.* Refer to Section 4.1: *Global Configuration Window* on page 72.

If no server-specific log policy is defined for the server, the server uses the System Default Logging Policy and this button is inactive. The values from the System Default Logging Policy are displayed in this window.

To define a server-specific log policy, alter the fields in this window as desired and press the **Add** button. A new log policy will be created for this server and the **Use Default Log Policy** button will become active. The new log policy will also be added to the *Logging Policies* list window.

To remove the server-specific log policy, press the **Use Default Log Policy** button. The button will become inactive again and the values of the System Default Logging Policy will be displayed in the window. The server-specific log policy will be deleted from the system and from the *Logging Policies* window.

To modify the server-specific log policy, alter the fields in this window as desired and press the **Update** button.

5.1.1. Core Server Specific Configuration

A separate Core Server must be configured for each storage subsystem.

The Specific tab of the *Core Server Configuration* window allows you to view and update the type-specific configuration for the Core Server.

COS Change Retry Limit, **Tape Dismount Delay**, **Tape Handoff Delay**, **PVL Max Connection Wait**, **Fragment Trim Limit** and **Fragment Smallest Block** can be changed in the Core Server while the server is running by changing the value on this screen, updating the metadata, then re-initializing the appropriate Core Server. The Core Server re-reads the metadata and changes its internal settings. The changes take effect the next time the settings are used by the server. See Section 5.2.2: *Reinitializing a Server* on page 154 for more information.

Changes made to the rest of the settings on this screen take effect the next time the server is started.

Field Descriptions

Root Fileset Name. Core Servers can create and support multiple filesets, but an initial fileset is required for each Core Server. The **Root Fileset Name** designates which of these filesets will be used by the Core Server to resolve the pathname "/" in the subsystem. Other filesets served by this Core Server, or from other Core Servers in the HPSS system may be joined to this fileset by junctions.

Root Fileset ID. The Fileset ID of the fileset named in **Root Fileset Name**.

Maximum Open Bitfiles. The maximum number of bitfiles that can be open simultaneously.

Maximum Active I/O Requests. The maximum number of simultaneous I/O requests allowed.

Maximum Active Copy Requests. The maximum number of simultaneous copy requests allowed.

COS Change Retry Limit. This is the maximum number of attempts that will be made to change the class of service of a file. If set to 0, COS change will continue to be attempted until it is successful. If a positive value is provided, the COS change request will be dropped after it has failed the configured number of times.

Tape Dismount Delay (seconds). The amount of time, in seconds, a mounted tape volume will remain idle before being dismounted by the Core Server. Larger values may reduce undesirable tape dismount/remount events, at the expense of lower tape drive utilization.

Tape Handoff Delay (seconds). The amount of time, in seconds, a mounted tape will be held in a client's session before become eligible to be handed off to another client that wishes to use the tape.

PVL Max Connection Wait (seconds). The amount of time, in seconds, the Core Server will wait to connect to the PVL before declaring an error in pending PVL jobs.

Fragment Trim Limit (clusters). Fragment Trim Limit sets the lower limit of the number of clusters that will be trimmed from a disk segment extent, when the extent turns out to be longer than the needed length. Larger values tend to reduce disk fragmentation, but at the expense of increased slack space.

Fragment Smallest Block. Fragment Smallest Block sets the boundary used to determine where to remove excess clusters from allocations that have excess space. The smallest block returned to the free space map from the excess space at the end of an extent will be this size or larger.

COS Copy to disk. If ON, all copy operations associated with COS changes should be directed to disk if the hierarchy has a disk storage class at its top level. If OFF, COS changes are not automatically copied to the disk level. If there is a tape level in the hierarchy, they will go there instead.

5.1.1.1. Additional Core Server Configuration

Certain aspects of the operation of the Core Server can be controlled by setting environment variables.

The server uses built-in default values for these settings, but if the environment variables can be found in the server's environment, the server uses those values. The following is a list of the names of the variables and the aspects of the server's operation they control. Since the environment is common to all sub-systems, all Core Servers in an HPSS installation are subject to these values.

HPSS_MAX_NUM_OPEN_FILES_FOR_AGGR – Adjusts the maximum number of files that may be open by the Core Server when it is assembling migration file aggregates. If this variable is not present in the environment, the server's default value is 20000.

HPSS_SPEC_USER_LIST_LENGTH – Adjusts the length of the Core Server's internal list of users who have delete only permission in AUTHZACL table. If this variable is not present in the environment, the server's default value is 8.

HPSS_RESTRICTED_USER_FILE – Contains the pathname of a file that contains the list of restricted users. If this variable is not present in the environment, the server does not implement this feature.

HPSS_CORE_REPACK_OUTPUT – Enables or disables the repack output segregation feature. If this environment variable is not present in the environment, or has the value “on”, files moved from one tape virtual volume to another by **repack** will be written to tapes that contain only files written by the **repack**. If this environment variable has the value “off”, files moved from one tape virtual volume to another by **repack** may mixed on output tapes with files written by the migration subsystem, or directly to tape by users.

HPSS_CORE_SEG_CACHE_IDLE_TIME – Adjusts the time, in seconds, a disk or tape storage segment can remain idle in the Core Server's in-memory cache before being purged. If this variable is not present in the environment, the server's default value is 60 seconds.

HPSS_CORE_TAPE_CACHE_IDLE_TIME – Adjusts the time, in seconds, a tape's in-memory cache entries can remain idle before being purged. This value controls the tape storage map, VV and PV caches. If this variable is not present in the environment, the server's default value is 300.

HPSS_CORE_LARGE_SEG_THRESHOLD – Controls the way tape storage space is allocated when large tape storage segments are “repacked” from one tape virtual volume to another. The value of this environment variable is a time, measured in seconds. If the segment to be moved would take longer to move than this time threshold, then the segment is moved to a tape VV that apparently has sufficient free space. If the segment will take less time to move than this threshold, then the segment is moved to any available tape, subject to the rules defined by **HPSS_CORE_REPACK_OUTPUT**. The server's default value is 30 seconds. The server uses the Transfer Rate information from the Storage Class when estimating the time a segment will take to move.

5.1.2. Gatekeeper Specific Configuration

To use a Gatekeeper for Gatekeeping Services the Gatekeeper must be configured into one or more Storage Subsystems (see Section 4.2.3: *Storage Subsystem Configuration Window* on page 76). To associate a Gatekeeper with a storage subsystem, the Gatekeeper must be selected on the Storage Subsystem Configuration window.

To use the Gatekeeper for Account Validation Services, Account Validation must be enabled in the Accounting Policy (see Section 13.2: *Accounting* on page 330). If Account Validation is configured to be ON, then at least one Gatekeeper must be configured even if the site does not wish to do gatekeeping. In this case, the Gatekeeper will only be used for validating accounts. Otherwise a Gatekeeper may be used for both gatekeeping and account validation. If multiple Gatekeepers are configured, then any Gatekeeper may be contacted for account validation requests.

Note: If a Gatekeeper is configured, then it will either need to be running or marked non-executable for HPSS Client API requests to succeed in the Core Server (even if no Gatekeeping nor Account Validation is occurring); this is due to the HPSS Client API performing internal accounting initialization.

For a detailed description of the Gatekeeper, please refer to the *HPSS Installation Guide*, Section 3.7.3: *Gatekeeper*.

Field Descriptions

Default Wait Time. The number of seconds the client must wait before retrying a request. This value must be greater than zero and is used if the Gatekeeping Site Interface returns a wait time of zero for the create, open, or stage request being retried.

Site Policy Pathname (UNIX). The contents of this file will be defined by the site and should be coordinated with the Gatekeeping Site Interface . For example, a site may want to define the types of requests being monitored, the maximum number of opens per user, and/or a path name to a log file. The Site Policy Pathname will be passed to the gk_site_Init routine upon Gatekeeper startup. If a site wishes to make use of the Site Policy Pathname file, then the site will need to add code to the Gatekeeping Site Interface library to read the Site Policy Pathname file and interpret the data accordingly. Refer to the *HPSS Programmer's Reference,* Chapter 4: *Site Interfaces* for details on the site's interface library.

5.1.3. Location Server Additional Configuration

The Location Server does not have a specific server configuration in the Server Configuration window. However, some additional configuration outside SSM is required whenever a Location Server is added or modified. Each client application which accesses HPSS must first contact the Location Server using the Location Servers's RPC endpoints, which must be listed in the /var/hpss/etc/ep.conf file to be available to client applications. The hpss_bld_ep utility reads the Location Server configuration and creates the ep.conf file. Refer to the *HPSS Installation Guide,* Appendix F: */var/hpss files* for more information on the ep.conf file.

The hpss_bld_ep utility must only run on the HPSS root subsystem machine. To invoke the utility:

```
% /opt/hpss/config/hpss_bld_ep -v -f /var/hpss/etc/ep.conf -r
<Unix/LDAP Realm Name> -c <schema name> -d <DB2 configuration
database> add
```

For example:

```
% /opt/hpss/config/hpss_bld_ep -v -f /var/hpss/etc/ep.conf -r
hpss.acme.com -c hpss -d cfg add
```

5.1.4. Log Client Specific Configuration

This window controls the local log settings that will be in effect for the node on which this Log Client runs.

Field Description

Client Port. The port number for communication between the Log Client and the HPSS Servers. The default value is 8101. Ensure that the specified port is not being used by other applications. The port number must be a different number than the Daemon Port used by the Log Daemon.

Maximum Local Log Size. The maximum size in bytes of the local log file. Once this size is reached, the log will be reused in a wraparound fashion. The default value is 5,242,880 (5 MB). The local log is not archived.

Local Logfile (UNIX). The fully qualified path name of the local log file. The default value is **/var/hpss/log/local.log**. If Local Logfile is specified in the **Log Messages To** field, those messages sent to this instance of the Log Client will be formatted and written to the designated file name. The specified file name will contain formatted messages only from HPSS applications running on the node on which this instance of the Log Client is running. This option is provided as a convenience feature. All HPSS messages will be written to a central log if Log Daemon is specified in the **Log Messages To** field.

Log Messages To. If neither Local Logfile nor Syslog is specified, no local logging will occur. If Log Daemon is not specified, messages from HPSS processes on the same node as this Log Client will not be written to the central log. The Syslog option should be used with care since the syslog file will grow without bound until it is deleted/truncated, or until the file system runs out of space. The following check boxes describe where the log messages will go.

- **Log Daemon.** If checked, messages are forwarded to the HPSS Log Daemon to be recorded in a central logfile.
- **Local Logfile.** If checked, messages are formatted and recorded in a local log file. This should not be checked if Stdout is also checked.
- **Syslog.** If checked, messages are formatted and sent to the host's syslog daemon.
- **Stdout.** If checked, messages are formatted and sent to the Log Client's standard output. This should not be checked if Local Logfile is also checked.

Related Information

Section 9.5: *Managing Local Logging* on page 301.

5.1.1. Log Daemon Specific Configuration

This window controls configuration of the log daemon and the central HPSS log.

Field Descriptions

Log File Maximum Size. The maximum size in bytes of each central log file. The default value is 5,242,880 (5 MB). Once the maximum size is reached, logging will switch to a second log file. The log file that filled up will then be archived to an HPSS file if the Archive Flag is on. Since two log files are allocated to accommodate log switching and log file archiving, the amount of space used by logging will be twice the specified value.

Switch Logfiles. A flag that indicates whether a switch to the second log file should be performed if the archive of the second log file has not yet completed. Use the default value of Always. Although a log may not get archived, the alternative may be a loss of logged messages to the current log from running servers.

Daemon Port. The port number for communication between the Log Client and the Log Daemon. The default value is 8100. Ensure that the port number assigned does not conflict with any other application on the node. The port value must be a different value than the Client Port number in the Log Client configuration entries.

Archive Logfiles. A flag that indicates whether log files should be archived when they fill.

Archive Class of Service. When a log file exceeds Log File Maximum Size, it will be archived to this class of service if the Archive Logfiles flag is set.

Log Directory. The name of the UNIX directory in which log files are stored. The default value is /var/hpss/log.

5.1.2. Migration/Purge Server (MPS) Specific Configuration

This tab of the *Migration Purge Server Configuration* window allows you to update/view the type-specific configuration parameters for a Migration/Purge Server (MPS).

As with any other part of the server configuration, updates made here are written directly to metadata with no involvement by the server. The server, if already running, must be recycled in order for the changes to become effective.

Field Descriptions

Storage Class Update Interval (seconds). The interval that indicates how often the MPS will query the Core Server in its subsystem to get the latest storage class statistics. This is also the interval the MPS uses to check whether it needs to initiate a purge operation on the storage class based on the associated purge policy. The valid range for this field is 10-600.

Maximum Volumes for Whole File Migration. The maximum number of tape virtual volumes that will be mounted for any tape storage class using the Migrate Volumes and Whole Files option. Migration of other storage classes, end user stage requests, and many other factors should be considered when deciding on this value.

Core Server API Failures Allowed. The maximum number of consecutive Core Server API failures allowed before MPS aborts a migration or purge run. This applies to both disk and tape migration as well as disk purge. If this limit is reached during a disk migration run, migration skips to the next hierarchy to which that disk storage class belongs. Once the run has attempted to migrate all such hierarchies the run will end. If this limit is reached during a disk purge or tape migration run, the migration or purge run aborts.

MPS Report File Name (path/prefix). A prefix string used by the MPS to construct a migration report file name. The full file name will consist of this string with a date string and subsystem ID appended to it. If this field is left blank, no migration reports will be generated. If a usable path prefix is not specified, the location of the migration report files may be unpredictable. A new MPS report file is started every 24 hours.

5.1.3. Mover Specific Configuration

5.1.3.1. Mover Specific Configuration Window

This tab of the *Mover Configuration* window allows you to update and view the type-specific configuration for the Mover.

The server must be recycled in order for changes to become effective.

The Mover cannot be configured entirely from the SSM window. See Section 5.1.3.1: *Additional Mover Configuration* on page 104, for further necessary configuration.

Field Descriptions

Encryption Key. An encryption key used to secure the Mover's interface with the Core Server. A specific value may be entered in the field or a random value may be generated by clicking on the **Generate** button. The default value is 0. A non-zero value must be configured to allow client access to the Mover's data interface. See Section 5.1.3.1: *Additional Mover Configuration* on page 104, for further information about the encryption key.

Buffer Size. The buffer size used for double buffering during data transfers. The default buffer size is set to 1,048,576 bytes (1 MB). This value should be tuned based on device and networking configuration and usage. The trade-off for this value is that large buffer sizes will use more system memory and may be inefficient for small transfers (e.g., if the Mover buffer size is 4MB, but client requests are 512KB, the Mover will not achieve any double buffering benefit because the entire amount of the transfer fits in one Mover buffer). A smaller buffer size will cause device and network I/O to be interrupted more often, usually resulting in reduced throughput rates for all but the smallest transfers.

The minimum Mover buffer size is the size of smallest block size for any device the Mover will handle. The maximum value will be bounded by the available system memory and the number of concurrent Mover requests anticipated.

TCP Path Name. The pathname of the Mover TCP/IP listen executable. The default value is "**/opt/hpss/bin/hpss_mvr_tcp**". The TCP Movers currently supported are listed below. All TCP Movers support common disk/tape interfaces, TCP/IP, and shared memory data transfers.

The hpss_mvr_ssd executable provides support for the IBM SCSI Tape Device Driver.

Table 4. Mover TCP Pathname Options

Name	**Options Supported**
hpss_mvr_tcp	Standard disk/tape devices – tape devices that use the Native SCSI device driver
hpss_mvr_ssd	Disk and SCSI 3590/3590E/3590H/3580, as well as tape device that uses IBM Atape device driver
hpss_mvr_dd2	Disk and Ampex DST-312 – tape devices that use the DD2 device driver

Hostname. The name of the host interface used for Mover control communication. This must be a valid host name for one of the network interfaces on the Mover node. The **Execute Hostname** of the *Core Server Configuration* window is used as the default. If the Mover is running remotely, this field must correspond to a network interface on the remote node, whereas the **Execute Hostname** set in the Mover's basic configuration corresponds to a network interface on which the administrative portion of the Mover runs.

Data Hostname. The host network interface name to be used by the Mover when transferring data. This must be a valid host name for one of the network interfaces on the Mover node. The default value is the **Execute Hosname** of the *Core Server Configuration* window. This field must correspond to a network interface on the remote node.

TCP Port. The TCP/IP port number used by the administrative interface portion of the Mover for the TCP/IP listen process to receive connections. It should be a value over 5000. The default value is 5001. The port number must be unique for processes running on the same node.

The Mover will internally use the port one greater than the one configured in this field on the remote node during initialization (e.g., if 5001 is entered, port 5002 is used on the remote node); therefore available port ranges on both nodes must be taken into consideration when selecting this value.

Port Range Start. The beginning of a range of local TCP/IP port numbers to be used by the Mover when connecting to clients. Valid values are zero or any valid TCP port number to which the Mover may bind (that is less than or equal to the value of Port Range End). The default value is 0. Use of particular port ranges may be required by some sites for communication across a firewall. If this value is set to zero, the operating system will select the port number; otherwise the Mover selects a local port number between Port Range Start and Port Range End (inclusive). If non-zero, this field must be less than or equal to Port

Range End. If this field is zero, **Port Range End** field must also be zero.

Port Range End. Used in conjunction with **Port Range Start** (See above). Valid values are zero or any TCP port number to which the Mover may bind (that is greater than or equal to the value of **Port Range Start**). The default value is 0. If non-zero, this field must be equal or greater than **Port Range Start**. If this field is zero, **Port Range Start** field must also be zero.

5.1.3.1. Additional Mover Configuration

The part of the Mover that handles accessing configuration metadata and servicing the Mover administrative interface runs on the core server platform , while the part of the Mover (with which the PVL and Core Servers communicate) that manages the storage devices and performs data transfers runs on a remote node. To support this mode of operation, there is some additional configuration which must be done on the remote node.

This additional configuration may be performed using mkhpss. See Section 5.1.3.1.1: *Setting Up Remote Movers with mkhpss* on page 108.

Alternately, the configuration may be performed manually using the instructions from these sections:

- 5.1.3.1.1: */etc/services, /etc/inetd.conf, and /etc/xinetd.d* on page 104.
- 5.1.3.1.2: *The Mover Encryption Key Files* on page 105.
- 5.1.3.1.3: */var/hpss/etc Files Required for Remote Mover* on page 106.
- 5.1.3.1.1: *System Configuration Parameters on IRIX, Solaris, and Linux* on page 106.

5.1.3.1.1. /etc/services, /etc/inetd.conf, and /etc/xinetd.d

To invoke the remote part of the Mover, **inetd** on the remote node is utilized to start the parent process when a connection is made to a port based on the Mover's type specific configuration (see Section 5.1.3: *Mover Specific Configuration* on page 102).

An entry must be added to the /etc/services file so that inetd will listen on the port to which the Mover client process will connect. The port number is one greater than that configured for the "TCP Listen Port" in the Mover's type specific configuration. For example, if the configured listen port is 5001, the following line must be added to /etc/services:

```
hpss_mvr1            5002/tcp
```

For non-Linux systems

An entry must be added to the /etc/inetd.conf file which defines which executable to run and the arguments to use when a connection is detected. The entry must use the service name specified for the Mover in /etc/services. The sole argument (other than the program name) is the pathname to a file that contains an ASCII representation of the configured encryption key for the Mover (see the next section for further details). For example, for the /etc/services entry added above, the corresponding /etc/inetd.conf entry would be:

```
hpss_mvr1 stream tcp nowait root /opt/hpss/bin/hpss_mvr_tcp
hpss_mvr_tcp /var/hpss/etc/mvr_ek
```

This will cause **inetd** to run the executable /opt/hpss/bin/hpss_mvr_tcp under the root user ID when a connection is detected on port 5002. The Mover process uses the /var/hpss/etc/mvr_ek file to read the encryption key that will be used to authenticate all connections made to this Mover.

After modifying the /etc/inetd.conf file, be sure to refresh the inetd daemon using the following commands:

```
% ps -ef | grep inetd
root  6450  3154   0   Apr 29      -  0:02 /usr/sbin/inetd
hpss 17852 59370   2 16:50:25 pts/18  0:00 grep inetd
% kill -1 6450
```

For Linux systems

A file must be added to the /etc/xinetd.d directory which defines which program to run and the arguments to use when a connection is detected. The file will be given the same name as the service name specified for this Mover in /etc/services. For example, for the /etc/services entry added above, the corresponding file /etc/xinetd.d/hpss_mvr1 would be created with the following contents:

```
service hpss_mvr1
{
     disable         = no
     socket_type     = stream
     protocol        = tcp
     wait            = no
     port            = 5002
     user            = root
     server          = /opt/hpss/bin/hpss_mvr_tcp
     server_args     = /var/hpss/etc/mvr_ek
}
```

The specified port will be one greater than the port listed as the TCP Listen Port in the Mover's type specific configuration. For example, the port value in the example corresponds to a Mover with a TCP Listen Port value of 5001.

This will cause **inetd** to run the executable /opt/hpss/bin/hpss_mvr_tcp under the root user ID when a connection is detected on port 5002. The Mover process will use the /var/hpss/etc/mvr_ek file to read the encryption key that will be used to authenticate all connections made to this Mover.

After modifying the file in /etc/xinetd.d, be sure to refresh xinetd daemon using the following commands:

```
% /sbin/service xinetd --full-restart
Stopping xinetd:                                [  OK  ]
Starting xinetd:                                [  OK  ]
```

5.1.3.1.2. The Mover Encryption Key Files

To authenticate access to the remote Mover processes, the encryption key configured in this Mover's specific configuration (see Section 5.1.3: *Mover Specific Configuration* on page 102) is read from a file accessible from the local file system. This file contains an ASCII representation of the encryption key. The pathname of the file is passed to the Mover executable as specified in either the /etc/inetd.conf or

/etc/xinetd.d file. For example, if the encryption key in the Mover's type specific configuration is **1234567890ABCDEF**, then the encryption key file (/var/hpss/etc/ek.mvr1) should contain:

```
0x12345678 0x90ABCDEF
```

5.1.3.1.3. /var/hpss/etc Files Required for Remote Mover

The Mover process on the remote machine requires access to the following files in /var/hpss/etc:

- auth.conf
- authz.conf
- env.conf
- ep.conf
- HPSS.conf
- ieee_802_addr
- site.conf

In addition, if the site is using Kererbos authentication, the following file is required:

- hpss.keytab

If the site is using Unix authentication, the following file is required:

- hpss.unix.keytab

If the site is using Unix authentication or authorization with local password files, the following files are required:

- passwd
- group

The ieee_802_addr file contains the hardware address of the host. It may be created using the /opt/hpss/config/get_802_addr program. This program requires no arguments. The remaining files should be copied from the HPSS root subsystem machine.

See the *HPSS Installation Guide*, Appendix F: */var/hpss files* for a description of each of these files.

5.1.3.1.1. System Configuration Parameters on IRIX, Solaris, and Linux

This section describes a number of system configuration parameters which may need to be modified for the successful operation of the remote Mover.

IRIX

IRIX system parameters which affect the remote Mover can be modified with the systune utility. It is likely that a system reboot will be required to make these parameters take effect. The following table defines the parameter names and minimum required values.

Table 1. IRIX System Parameters

Parameter Name	Minimum Value	Parameter Description
semmsl	512	Maximum number of semaphores per set
maxdmasz	513	Maximum DMA size (required for Ampex DST support)

Solaris

Solaris system parameters which affect the remote Mover can be modified by editing the /etc/system configuration file and rebooting the system. The following table defines the parameter names and minimum required values.

Note that the semmns and semmnu values should be increased if running more than one Mover on the Solaris machine (multiply the minimum value by the number of Movers to be run on that machine).

Table 2. Solaris System Parameters

Parameter Name	Minimum Value	Parameter Description
semsys:seminfo_semmns	1024	Maximum number of semaphores in system
semsys:seminfo_semmnu	512	Maximum number of undo structures in system
semsys:seminfo_semmsl	512	Maximum number of semaphores per set
shmsys:shminfo_shmmax	8388608	Maximum shared memory segment size (will allow up to a 2MB Mover buffer size)

Linux

Linux system parameters which affect the remote Mover are detailed in the following table and can be modified as follows.

To temporarily change **SEMMSL** and **SHMMAX**, use the **sysctl**(8) commands as shown in the example below (the values take effect immediately, but will not be preserved across reboots).

```
% /sbin/sysctl -w kernel.sem="512 32000 32 128"
% /sbin/sysctl -w kernel.shmmax="33554432"
```

Where "**512 32000 32 128**" are the values for **SEMMSL**, **SEMMNS**, **SEMOPM**, and **SEMNI** respectively (only change **SEMMSL**).

To make the changes permanent (across reboots), edit **sysctl.conf**(5) and add the following lines (preserving spaces).

```
# Increase SEMMSL and SHMMAX for HPSS Mover
kernel.sem = 512 32000 32 128
kernel.shmmax = 33554432
```

Note that the minimum **SEMMSL** value in the following table should be multiplied by the number of Movers running on the Linux machine.

Table 3. Linux System Parameters

Parameter Name	Minimum Value	Parameter Description
SEMMSL	512	Maximum number of semaphores per ID
SHMMAX	0x2000000	Maximum shared memory segment size (bytes)

5.1.3.1.1. Setting Up Remote Movers with mkhpss

The mkhpss utility may be used to copy the files needed for a remote Mover from the root subsystem machine, to create the files which may not be copied, to install the required files on the remote Mover machine, and to configure the inetd to run the remote Mover.

To create the necessary files, run mkhpss on the root subsystem machine. From the Root Subsystem Machine menu, select the Create Config Bundle option. From this window, set the desired name of the config file bundle and push the **Create Config Bundle** button.

To install the files and configure the inetd on the remote Mover machine, transfer the created config bundle to the remote Mover machine and run mkhpss on the remote Mover machine. From the Mover/Client Machine menu, select the Install Config Bundle item. On this window, set the approprate name of the bundle file and push the **Install Config Bundle** button.

Next, select the Mover/Client Machine menu Configuration submenu, and from this, the Security Services item. From this window, select the authentication mechanism (Kerberos or Unix) using the drop down menu of the **Authentication Mechanism** field. For Kerberos, complete the remaining fields on the window and push the **Configure Security Services** button to set up the required files. For Unix, there is nothing further to configure on this window.

Next, still under the Mover/Client Machine menu Configuration submenu, select the Other Services item. On this window, select the checkbox for "Configure any Movers for this machine". This checkbox instructs mkhpss to add the proper lines for the Movers to /etc/services. It also instructs it, depending upon the operating system type, either to modify the /etc/inetd.conf file or to add the appropriate configuration file to the /etc/xinetd.d directory to support the Movers. On this same window, select the checkbox for "Set any system parameters necessary for HPSS"; this configures the local system settings described in section 5.1.3.1.1: System Configuration Parameters on IRIX, Solaris, and Linux on page 106. At this time, there are no additional system parameters to set for AIX, so it is not actually necessary to select this checkbox for AIX hosts.

5.1.3.1.2. Mover Configuration to Support Local File Transfer

The Mover local file transfer (LFT) protocol allows the Movers to transfer data directly between a UNIX file system and HPSS.

Caveats:

- This transfer can only occur if both the Mover and the clients have direct access to the UNIX filesystem on which the file resides.
- If the multinode features of the Parallel FTP Client are used, the UNIX filesystem must be global

to all nodes or unexpected results may occur.

- The Mover must be built with the LFT option. This is the default option for all Movers. If not all Movers have been built with this option, clients must explicitly specify a class of service which is valid for a Mover supporting the local file transfer option.
- The Mover must be running as root. If there are other Movers running on the same node, they must also run as root to take advantage of the Mover-to-Mover shared memory data transfer.
- A new configuration file (**`/var/hpss/etc/hpss_mvr_localfilepath.conf`**) is required on all nodes running an enhanced Mover.

The configuration file contains a list of UNIX paths. These paths specify which files are allowed to be moved using this special data transfer protocol. Any UNIX file on the Mover node whose path starts with a path in the list of entries is allowed to be transferred using the special data protocol.

If the configuration file does not exist, then the SSM will issue a minor alarm when the Mover starts, and no UNIX files will be transferred with the special protocol.

Here is an example configuration file:

```
# This file contains Unix paths that can be accessed by the
# local Movers on this node. If the client wishes to make a
# local file transfer and the start of the client path matches
# any of the paths in this list then the transfer will proceed,
# otherwise the Mover will not transfer the file.
#
# The format of this file is simply a list of paths, one per
# line.
/gpfs
/local/globalfilesystem
```

In the above sample configuration, any file under the path, /gpfs or the path /local/globalfilesystem can be transferred using the special data protocol subject to the caveats specified above.

Extreme care should be taken when using this feature. The Mover will move data from any file in its list. If the UNIX file system is not global to all Mover nodes, the request will fail.

The following commands have been added to the HPSS Parallel FTP Clients to take advantage of these new features: lfget, lfput, mlfget, mlfput, lfappend. See the HPSS User's Guide for more details. The hpss_ReadList and hpss_WriteList Client APIs can also be used to utilize the local file transfer capability. See the HPSS Programmer's Reference - I/O Supplement for more details.

5.1.1. Physical Volume Repository (PVR) Specific Configuration

The PVR specific configuration entry can be created and managed using the *PVR Server Configuration* window. If changes are made while the server is running, the server must be recycled in order for the changes to become effective.

If you are configuring a PVR for StorageTek, IBM 3494/3584, ADIC AML, LTO, or generic SCSI: before proceeding with PVR configuration you should read the PVR-specific section (Section 5.1.1.1: STK PVR

Specific Configuration Window on page 120, Section 5.1.1.1: 3494 PVR Specific Configuration on page 111, Section 5.1.1.2: AML PVR Specific Configuration on page 113, and Section 5.1.1.2: SCSI PVR Specific Configuration Window on page 118). These sections provide additional vendor-specific advice on PVR/robot configuration.

The AML PVR is supported by special bid only.

5.1.1.1. Operator PVR Specific Configuration Window

Field Descriptions

Cartridge Capacity. The total number of cartridge slots in the library dedicated to this HPSS PVR. This may or may not be the total cartridge capacity of the library; a site might use part of the library for some other HPSS PVR or for some non-HPSS application. The PVR uses the **Cartridge Capacity** field and the **Cartridge Alarm Threshold** field to determine when to send an alarm that the total cartridge threshold has been exceeded.

Cartridge Alarm Threshold. The percentage of the **Cartridge Capacity** at which the PVR will send an alarm.

Shelf Tape Check-In Retry. The number of seconds the PVR will wait before asking the robot if a requested shelf tape has been checked-in. The PVR will continue checking at this interval until the tape is checked-in. This field applies only if the **Support Shelf Tape** checkbox is selected. The retry value must be 30 or greater.

Shelf Tape Check-In Alarm. The PVR will periodically log alarm messages when a requested shelf tape has not been checked-in. This field specifies the number of minutes between alarms. This field applies only if the **Support Shelf Tape** checkbox is selected. The alarm value must be 2 or greater.

Dismount Delay. When **Defer Dismounts** is checked, this value is used by the PVL to determine the number of minutes that dismounts are delayed after the last data access.

Characteristics. Flags for the PVR:

- **Defer Dismounts.** If ON, the PVL will delay the dismounting of a tape cartridge until the drive is required by another job or until the **Dismount Delay** time limit is exceeded.
- **Support Shelf Tape.** If ON, the PVR and the PVL will support the removal of cartridges from the tape library using the shelf_tape utility.

5.1.1.1. 3494 PVR Specific Configuration

5.1.1.1.1. 3494 PVR Specific Configuration Window

Field Descriptions

Cartridge Capacity. The total number of cartridge slots in the library dedicated to this HPSS PVR. This may or may not be the total cartridge capacity of the library; a site might use part of the library for some other HPSS PVR or for some non-HPSS application. The PVR uses the **Cartridge Capacity** field and the **Cartridge Alarm Threshold** field to determine when to send an alarm that the total cartridge threshold has been exceeded.

Cartridge Alarm Threshold. The percentage of the Cartridge Capacity at which the PVR will send an alarm.

Same Job on Controller, **Other Job on Controller**, & **Distance To Drive**. These values are used by the PVR when selecting a drive for a tape mount operation. The three values are essentially weights that are used to compute an overall score for each possible drive. After the score has been calculated, the drive with the lowest score is selected for the mount. If two or more drives tie for the lowest score, one drive is selected at random. The score is calculated as follows:

Score =

Weight 1 * Cartridges from this job mounted on this drive's controller +

Weight 2 * Cartridges from other jobs mounted on this drive's controller +

Weight 3 * Units of distance from the cartridge to the drive

This method has the effect of distributing a striped tape mount across as many controllers as possible for the best performance. It also will try to pick controllers that are currently driving a minimum number of tapes. So, in an environment with many tape drives per controller, the best performance will be achieved by minimizing the load on any one controller.

The **Distance To Drive** helps minimize mount times by mounting the tape in a physically close drive. For IBM robots, the time it takes to move a cartridge to a more distant drive is not significant, so the default value here is fairly low. The unit of distance for a 3494 library is one frame. All other things being equal, the tape will be mounted in the closest drive. However, if the closest drive is attached to a heavily used controller, then a more distant drive will be selected.

Shelf Tape Check-In Retry. The number of seconds the PVR will wait before asking the robot if a requested shelf tape has been checked-in. The PVR will continue checking at this interval until the tape is checked-in. This field applies only if the **Support Shelf Tape** checkbox is selected. The retry value must be 30 or greater.

Shelf Tape Check-In Alarm. The PVR will periodically log alarm messages when a requested shelf tape has not been checked-in. This field specifies the number of minutes between alarms. This field applies only if the **Support Shelf Tape** checkbox is selected. The alarm value must be 2 or greater.

Dismount Delay. When **Defer Dismounts** is checked, this value is used by the PVL to determine the number of minutes that dismounts are delayed after the last data access.

Retry Mount Time Limit. The default value for this field is -1. When the default value (-1) is used, if an error is encountered during a PVR mount operation, the mount will pend and be retried every 5 minutes. Setting a value in this field will change the mount behavior to periodically retry the mount until the specified time limit is exceeded. Once exceeded, an error is generated and the mount request is canceled. If the mount request would have resulted in a write operation, the error returned will cause the Core Server to set the **VV Condition** of the associated tape volume to DOWN. Once in DOWN state, the volume will no longer be available for read or write operations. For further information about the Core Server VV Condition, see Section 4.5.4.2: *Core Server Tape Volume Information Window* on page 471.

Drive Error Limit. This field is intended to be used in conjunction with the PVR Server **Retry Mount Time Limit**. If the number of consecutive mount errors which occur to any drive in this PVR equal or exceed this value, the drive is automatically locked by the PVL. The only mount errors that apply are those set through the **Retry Mount Time Limit** mechanism. The **Drive Error Count** field in the *PVL Drive Information* records the number of consecutive errors on a drive by drive basis. To turn off the automatic drive disable feature, set the **Drive Error Limit** to 0 or -1. Since this field is employed by the PVL, changing its value will not change drive disable behavior until the PVL is recycled.

Characteristics. Flags for the PVR:ismounts. If ON, the PVL will delay the dismounting of a tape cartridge until the drive is required by another job or until the Dismount Delay time limit is exceeded.

- **Support Shelf Tape.** If ON, the PVR and the PVL will support the removal of cartridges from the tape library using the shelf_tape utility.

Command Device. The name of the device that the PVR can use to send commands to the robot. For AIX systems, this is generally **/dev/lmcp0**. For Linux systems, use the symbolic library name defined in **/etc/ibmatl.conf**. The environment variable **HPSS_3494_COMMAND_DEVICE** will override the value entered in this field. Only use the environment variable when using only one 3494.

Async Device. The name of the device that the PVR can use to receive replies from the 3494 robot. For AIX systems, this is generally **/dev/lmcp1**. For Linux systems, use the symbolic library name defined in **/etc/ibmatl.conf**. For TTY and LAN-attached robots, the **Async Device** may be the same as the **Command Device**. The environment variable **HPSS_3494_ASYNC _DEVICE** will override the value entered in this field. Only use the environment variable when using only one 3494.

5.1.1.1.1. 3494 PVR Additional Information

Configuration Requirements

When configuring an HPSS PVR to manage an IBM robot, the Drive Address configuration entries correspond to the device number of the drive. Determine the device number by running the HPSS tool GetESANumbers for each tape drive in the robot.

The HPSS PVR requires one or two LMCP device special files to access the robot. For AIX systems, these files are usually named **/dev/lmcpX** and can be created using the System Management Interface Tool (SMIT). The files can be defined to be either a Command Port or an Asynchronous (Async) Port. The HPSS PVR requires both. By default, the PVR will expect the Command and Async ports to be named **/dev/lmcp0** and **/dev/lmcp1** respectively. Control connections must be made prior to configuration of the **/dev/lmcpX** devices or undefined errors may result. For Linux systems, the symbolic library name defined in **/etc/ibmatl.conf** (e.g., 3494a) should be used.

For RS-232 and Ethernet connected robots, the device special files support both command and async capabilities. If only one device special file is created, the environment variables or configuration should be set so that both the command and async ports point to that one special file. It is also possible to create two device special files and arbitrarily select one to be the command port and the other to be the async port. The PVR can then be configured to recognize the ports as described above.

HPSS can share an IBM robot with other tape management systems. If a robot is shared, care must be taken to make sure that a drive is not used by any other tape management system while that drive is configured as unlocked in the HPSS PVL. This is important because HPSS periodically polls all of its unlocked drives even if they are not currently mounted. The two LMCP device special files (or possibly one file in the case of an RS-232 or Ethernet controlled robot) are not available to other tape management programs. Additional device special files may be created for other programs. Other programs can send commands to the robot at the same time as HPSS through the additional device special file.

If the robot is placed in pause mode by an operator, an alarm will appear on the HPSS operator window. All subsequent robot operations will silently be suspended until the robot is put back in automatic mode.

5.1.1.2. AML PVR Specific Configuration

5.1.1.2.1. AML PVR Specific Configuration Window

Field Descriptions

Cartridge Capacity. The total number of cartridge slots in the library dedicated to this HPSS PVR. This may or may not be the total cartridge capacity of the library; a site might use part of the library for some other HPSS PVR or for some non-HPSS application. The PVR uses the **Cartridge Capacity** field and the **Cartridge Alarm Threshold** field to determine when to send an alarm that the total cartridge threshold has been exceeded.

Cartridge Alarm Threshold. The percentage of the Cartridge Capacity at which the PVR will send an alarm.

Same Job on Controller, **Other Job on Controller**, & **Distance To Drive**. These values are used by the PVR when selecting a drive for a tape mount operation. The three values are essentially weights that are used to compute an overall score for each possible drive. After the score has been calculated, the drive with the lowest score is selected for the mount. If two or more drives tie for the lowest score, one drive is selected at random. The score is calculated as follows:

Score =

Weight 1 * Cartridges from this job mounted on this drive's controller +

Weight 2 * Cartridges from other jobs mounted on this drive's controller +

Weight 3 * Units of distance from the cartridge to the drive

This method has the effect of distributing a striped tape mount across as many controllers as possible for the best performance. It also will try to pick controllers that are currently driving a minimum number of tapes. So, in an environment with many tape drives per controller, the best performance will be achieved by minimizing the load on any one controller.

The **Distance To Drive** helps minimize mount times by mounting the tape in a physically close drive. All other things being equal, the tape will be mounted in the closest drive. However, if the closest drive is attached to a heavily used controller, then a more distant drive will be selected.

Shelf Tape Check-In Retry. The number of seconds the PVR will wait before asking the robot if a requested shelf tape has been checked-in. The PVR will continue checking at this interval until the tape is checked-in. This field applies only if the **Support Shelf Tape** checkbox is selected. The retry value must be 30 or greater.

Shelf Tape Check-In Alarm. The PVR will periodically log alarm messages when a requested shelf tape has not been checked-in. This field specifies the number of minutes between alarms. This field applies only if the **Support Shelf Tape** checkbox is selected. The alarm value must be 2 or greater.

Dismount Delay. When **Defer Dismounts** is checked, this value is used by the PVL to determine the number of minutes that dismounts are delayed after the last data access.

Retry Mount Time Limit. The default value for this field is -1. When the default value (-1) is used, if an error is encountered during a PVR mount operation, the mount will pend and be retried every 5 minutes. Setting a value in this field will change the mount behavior to periodically retry the mount until the specified time limit is exceeded. Once exceeded, an error is generated and the mount request is canceled. If the mount request would have resulted in a write operation, the error returned will cause the Core Server to set the **VV Condition** of the associated tape volume to DOWN. Once in DOWN state, the volume will no longer be available for read or write operations. For further information about the Core Server VV Condition, see Section 4.5.4.2: *Core Server Tape Volume Information Window* on page 271.

Drive Error Limit. This field is intended to be used in conjunction with the PVR Server **Retry Mount Time Limit**. If the number of consecutive mount errors which occur to any drive in this PVR equal or exceed this value, the drive is automatically locked by the PVL. The only mount errors that apply are those set through the **Retry Mount Time Limit** mechanism. The **Drive Error Count** field in the *PVL Drive Information* records the number of consecutive errors on a drive by drive basis. To turn off the automatic drive disable feature, set the **Drive Error Limit** to 0 or -1. Since this field is employed by the PVL, changing its value will not change drive disable behavior until the PVL is recycled.

Characteristics. Flags for the PVR:

- **Defer Dismounts.** If ON, the PVL will delay the dismounting of a tape cartridge until the drive is required by another job or until the **Dismount Delay** time limit is exceeded.
- **Support Shelf Tape.** If ON, the PVR and the PVL will support the removal of cartridges from the tape library using the shelf_tape utility.

Client Name . The name of the client requesting authorization from the Distributed Automated Media Library Server. DAS software, which executes on the OS/2 controller PC, allows different clients to control the AML robotics system. DAS uses the client name to determine the access permission of the requesting client to the AML's storage positions, drives, and Insert/Eject units. Access configurations for clients are set in the configuration file C:\DAS\ETC\CONFIG on the OS/2 PC. The client name can be up to 64 alphanumeric characters in length and is case sensitive.

Server Name. TCP/IP host name or IP address of the AML OS/2-PC DAS server. This value must be defined in the network domain server and must be resolvable during DAS start. The server name is set in the configuration file C:\CONFIG.SYS on the OS/2 PC. The server name can be up to 64 alphanumeric characters long and can include up to six dots ('.').

5.1.1.2.1. AML PVR Additional Information

The AML PVR is supported by special bid only.

ADIC Automatic Media Library Storage Systems Information

HPSS can share an AML robot with other tape management systems. If a robot is shared, care must be taken to make sure that a drive is not used by any other tape management system while that drive is configured as unlocked in HPSS. This is important because HPSS can be configured to periodically poll all of its unlocked drives even if they are not currently mounted or in use by HPSS. If a drive is being used by another tape management system, it must be configured as locked in HPSS. For robots that have more than one arm (such as the AML/2), users should configure the PVL Drive Information/Controller ID of each drive depending on which arm is servicing it.

User needs to set the Server Name and Client Name, which are case sensitive, in the AML PVR Server Configuration panel to establish the connectivity between the HPSS software and the OS/2 controlling the robot. The Server Name is the name of the controller associated with the TCP/IP address, as defined in the TCP/IP HOST file, and the Client Name is the name of the OS/2 administrator client as defined in the DAS configuration.

Additionally users must configure the Insert/Eject ports for the AML PVR using the configuration files /var/hpss/etc/AML_EjectPort.conf and /var/hpss/etc/AML_InsertPort.conf. The reasons are that the AML robot can have multiple insert and eject ports, which have the capability to handle different media types. These two configuration files in conjunction with the AMU AMS configuration files specify which Insert/Eject areas or bins the tapes should be placed in for insertion into the archive and where they are ejected to when the HPSS export command is used.

Note that if multiple E/I/F bins are used for exporting cartridges, users must set the environment variable DAS_EJECTAREAFULL=1 on the DAS Server PC by doing the following:

1. Edit the config.sys
2. Add a new line:

```
set DAS_EJECTAREAFULL=1
```

Starting the DAS Server Processes

In order for the HPSS AML PVR to communicate with the AML robot, it is required that the DAS Server process be running.

Once the DAS, TCP/IP and AML configuration have been completed, the DAS server is initialized and started with the following steps:

1. Make sure the AMU archive management software is running and the hostname is resolved,
2. Select an OS/2 window from the Desktop and change the directory to C:\DAS,

   ```
   C:> cd \das
   ```
3. At the prompt, type tcpstart and make sure that TCP/IP gets configured and that the port mapper program is started,

   ```
   C:\das> tcpstart
   ```
4. Type dasstart to start the DAS server

   ```
   C:\das> dasstart
   ```

In most cases, a startup file like the following can be used to automatically start the DAS processor:

```
call tcpstart
das\tools\os2sleep 20
CD \AMU
START CON
START KRN
cd \das
tools\os2sleep 20
call dasstart
cd bin
start "DAS/2 AmuClient"
exit
```

5.1.1.1. LTO PVR Specific Configuration

5.1.1.1.1. LTO PVR Specific Configuration Window

Field Descriptions

Cartridge Capacity. The total number of cartridge slots in the library dedicated to this HPSS PVR. This may or may not be the total cartridge capacity of the library; a site might use part of the library for some other HPSS PVR or for some non-HPSS application. The PVR uses the **Cartridge Capacity** field and the **Cartridge Alarm Threshold** field to determine when to send an alarm that the total cartridge threshold has been exceeded.

Cartridge Alarm Threshold. The percentage of the Cartridge Capacity at which the PVR will send an alarm.

Same Job on Controller, **Other Job on Controller**, & **Distance To Drive**. These values are used by the PVR when selecting a drive for a tape mount operation. The three values are essentially weights that are used to compute an overall score for each possible drive. After the score has been calculated, the drive with the lowest score is selected for the mount. If two or more drives tie for the lowest score, one drive is selected at random. The score is calculated as follows:

Score =

Weight 1 * Cartridges from this job mounted on this drive's controller +

Weight 2 * Cartridges from other jobs mounted on this drive's controller +

Weight 3 * Units of distance from the cartridge to the drive

This method has the effect of distributing a striped tape mount across as many controllers as possible for the best performance. It also will try to pick controllers that are currently driving a minimum number of tapes. So, in an environment with many tape drives per controller, the best performance will be achieved by minimizing the load on any one controller.

The **Distance To Drive** helps minimize mount times by mounting the tape in a physically close drive. For IBM robots, the time it takes to move a cartridge to a more distant drive is not significant, so the default value here is fairly low. All other things being equal, the tape will be mounted in the closest drive. However, if the closest drive is attached to a heavily used controller, then a more distant drive will be selected.

Shelf Tape Check-In Retry. The number of seconds the PVR will wait before asking the robot if a requested shelf tape has been checked-in. The PVR will continue checking at this interval until the tape is checked-in. This field applies only if the **Support Shelf Tape** checkbox is selected. The retry value must be 30 or greater.

Shelf Tape Check-In Alarm. The PVR will periodically log alarm messages when a requested shelf tape has not been checked-in. This field specifies the number of minutes between alarms. This field applies only if the **Support Shelf Tape** checkbox is selected. The alarm value must be 2 or greater.

Dismount Delay. When **Defer Dismounts** is checked, this value is used by the PVL to determine the number of minutes that dismounts are delayed after the last data access.

Retry Mount Time Limit. The default value for this field is -1. When the default value (-1) is used, if an error is encountered during a PVR mount operation, the mount will pend and be retried every 5 minutes. Setting a value in this field will change the mount behavior to periodically retry the mount until the specified time limit is exceeded. Once exceeded, an error is generated and the mount request is canceled. If the mount request would have resulted in a write operation, the error returned will cause the Core Server to set the **VV Condition** of the associated tape volume to DOWN. Once in DOWN state, the volume will no longer be available for read or write operations. For further information about the Core Server VV Condition, see Section 4.5.4.2: *Core Server Tape Volume Information Window* on page 271.

Drive Error Limit. This field is intended to be used in conjunction with the PVR Server **Retry Mount Time Limit**. If the number of consecutive mount errors which occur to any drive in this PVR equal or exceed this value, the drive is automatically locked by the PVL. The only mount errors that apply are those set through the **Retry Mount Time Limit** mechanism. The **Drive Error Count** field in the *PVL Drive Information* records the number of consecutive errors on a drive by drive basis. To turn off the automatic drive disable feature, set the **Drive Error Limit** to 0 or -1. Since this field is employed by the PVL, changing its value will not change drive disable behavior until the PVL is recycled.

Characteristics. Flags for the PVR:

- **Defer Dismounts.** If ON, the PVL will delay the dismounting of a tape cartridge until the drive is required by another job or until the **Dismount Delay** time limit is exceeded.

- **Support Shelf Tape.** If ON, the PVR and the PVL will support the removal of cartridges from the tape library using the shelf_tape utility.

Command Device. The name of the device that the PVR can use to send commands to the robot. For AIX systems, this is generally **/dev/smc0**. For Linux systems, use the symbolic library name defined in **/etc/ibmatl.conf**.

5.1.1.1.1. LTO PVR Additional Information

The LTO PVR must run on the same node that has the Atape interface and this node must have a direct SCSI connection to the library. Please note that the control channel for the library is shared with the data path for the first drive in the library over the same SCSI interface.

The 3584 Library can be partitioned into multiple logical libraries which can each be controlled by a separate PVR. This is useful if you have a large library, since you can allocate a set of tapes to a specific set of drives under a logical library.

While HPSS can theoretically share an IBM robot with other tape management systems, this is not recommended. If a robot is shared, care must be taken to ensure that a drive is not used by any other tape management system while that drive is unlocked in HPSS. This is important because HPSS periodically polls all of its unlocked drives even if they are not currently mounted. The SMC device special file is not available to other tape management programs. Additional device special files may be created for other programs.

The LTO PVR requires an SMC device special file to access the robot. For AIX systems, this file is usually named **/dev/smc0** and can be created using the System Management Interface Tool (SMIT) or cfgmgr.

5.1.1.2. SCSI PVR Specific Configuration Window

Field Descriptions

Cartridge Capacity. The total number of cartridge slots in the library dedicated to this HPSS PVR. This may or may not be the total cartridge capacity of the library; a site might use part of the library for some other HPSS PVR or for some non-HPSS application. The PVR uses the **Cartridge Capacity** field and the **Cartridge Alarm Threshold** field to determine when to send an alarm that the total cartridge threshold has been exceeded.

Cartridge Alarm Threshold. The percentage of the **Cartridge Capacity** at which the PVR will send an alarm.

Same Job on Controller, **Other Job on Controller**, & **Distance To Drive**. These values are used by the PVR when selecting a drive for a tape mount operation. The three values are essentially weights that are used to compute an overall score for each possible drive. After the score has been calculated, the drive with the lowest score is selected for the mount. If two or more drives tie for the lowest score, one drive is selected at random. The score is calculated as follows:

Score =

Weight 1 * Cartridges from this job mounted on this drive's controller +

Weight 2 * Cartridges from other jobs mounted on this drive's controller +

Weight 3 * Units of distance from the cartridge to the drive

This method has the effect of distributing a striped tape mount across as many controllers as possible for the best performance. It also will try to pick controllers that are currently driving a minimum number of tapes. So, in an environment with many tape drives per controller, the best performance will be achieved by minimizing the load on any one controller.

The **Distance To Drive** helps minimize mount times by mounting the tape in a physically close drive. All other things being equal, the tape will be mounted in the closest drive. However, if the closest drive is attached to a heavily used controller, then a more distant drive will be selected.

Shelf Tape Check-In Retry. The number of seconds the PVR will wait before asking the robot if a requested shelf tape has been checked-in. The PVR will continue checking at this interval until the tape is checked-in. This field applies only if the **Support Shelf Tape** checkbox is selected. The retry value must be 30 or greater.

Shelf Tape Check-In Alarm. The PVR will periodically log alarm messages when a requested shelf tape has not been checked-in. This field specifies the number of minutes between alarms. This field applies only if the **Support Shelf Tape** checkbox is selected. The alarm value must be 2 or greater.

Dismount Delay. When **Defer Dismounts** is checked, this value is used by the PVL to determine the number of minutes that dismounts are delayed after the last data access.

Retry Mount Time Limit. The default value for this field is -1. When the default value (-1) is used, if an error is encountered during a PVR mount operation, the mount will pend and be retried every 5 minutes. Setting a value in this field will change the mount behavior to periodically retry the mount until the specified time limit is exceeded. Once exceeded, an error is generated and the mount request is canceled. If the mount request would have resulted in a write operation, the error returned will cause the Core Server to set the **VV Condition** of the associated tape volume to DOWN. Once in DOWN state, the volume will no longer be available for read or write operations. For further information about the Core Server VV Condition, see Section 4.5.4.2: *Core Server Tape Volume Information Window* on page 271.

Drive Error Limit. This field is intended to be used in conjunction with the PVR Server **Retry Mount Time Limit**. If the number of consecutive mount errors which occur to any drive in this PVR equal or exceed this value, the drive is automatically locked by the PVL. The only mount errors that apply are those set through the **Retry Mount Time Limit** mechanism. The **Drive Error Count** field in the *PVL Drive Information* records the number of consecutive errors on a drive by drive basis. To turn off the automatic drive disable feature, set the **Drive Error Limit** to 0 or -1. Since this field is employed by the PVL, changing its value will not change drive disable behavior until the PVL is recycled.

Characteristics. Flags for the PVR:

- **Defer Dismounts.** If ON, the PVL will delay the dismounting of a tape cartridge until the drive is required by another job or until the **Dismount Delay** time limit is exceeded.
- **Support Shelf Tape.** If ON, the PVR and the PVL will support the removal of cartridges from the tape library using the shelf_tape utility.

- **Enforce Home Location.** If ON, the SCSI PVR will always try to dismount a mounted cart back to its home location. Otherwise, it will just use the first free slot. The scsi_home utility can be used to view and manipulate the home location values.

Serial Number. The serial number of the robot, obtained from **device_scan**. This serial number will allow the SCSI PVR to automatically look up all available control paths upon startup.

5.1.1.1. STK PVR Specific Configuration Window

Field Descriptions

Cartridge Capacity. The total number of cartridge slots in the library dedicated to this HPSS PVR. This may or may not be the total cartridge capacity of the library; a site might use part of the library for some other HPSS PVR or for some non-HPSS application. The PVR uses the **Cartridge Capacity** field and the **Cartridge Alarm Threshold** field to determine when to send an alarm that the total cartridge threshold has been exceeded.

Cartridge Alarm Threshold. The percentage of the **Cartridge Capacity** at which the PVR will send an alarm.

Same Job on Controller, **Other Job on Controller**, & **Distance To Drive**. These values are used by the PVR when selecting a drive for a tape mount operation. The three values are essentially weights that are used to compute an overall score for each possible drive. After the score has been calculated, the drive with the lowest score is selected for the mount. If two or more drives tie for the lowest score, one drive is selected at random. The score is calculated as follows:

Score =

Weight 1 * Cartridges from this job mounted on this drive's controller +

Weight 2 * Cartridges from other jobs mounted on this drive's controller +

Weight 3 * Units of distance from the cartridge to the drive

This method has the effect of distributing a striped tape mount across as many controllers as possible for the best performance. It also will try to pick controllers that are currently driving a minimum number of tapes. So, in an environment with many tape drives per controller, the best performance will be achieved by minimizing the load on any one controller.

The **Distance To Drive** helps minimize mount times by mounting the tape in a physically close drive. For STK drives, a unit of distance is from one Silo/LSM (Library Storage Module) to the next. This means that the tape must go through a pass-through port or elevator. This process is more time consuming. Therefore the default value for STK robots is a higher value which forces the tape to stay in the same Silo/LSM even if it means the drive selected is attached to a controller which is heavily used. All other things being equal, the tape will be mounted in the closest drive.

Shelf Tape Check-In Retry. The number of seconds the PVR will wait before asking the robot if a requested shelf tape has been checked-in. The PVR will continue checking at this interval until the tape is checked-in. This field applies only if the **Support Shelf Tape** checkbox is selected. The retry value must be 30 or greater.

Shelf Tape Check-In Alarm. The PVR will periodically log alarm messages when a requested shelf tape has not been checked-in. This field specifies the number of minutes between alarms. This field applies only if the **Support Shelf Tape** checkbox is selected. The alarm value must be 2 or greater.

Dismount Delay. When **Defer Dismounts** is checked, this value is used by the PVL to determine the number of minutes that dismounts are delayed after the last data access.

Retry Mount Time Limit. The default value for this field is -1. When the default value (-1) is used, if an error is encountered during a PVR mount operation, the mount will pend and be retried every 5 minutes. Setting a value in this field will change the mount behavior to periodically retry the mount until the specified time limit is exceeded. Once exceeded, an error is generated and the mount request is canceled. If the mount request would have resulted in a write operation, the error returned will cause the Core Server to set the **VV Condition** of the associated tape volume to DOWN. Once in DOWN state, the volume will no longer be available for read or write operations. For further information about the Core Server VV Condition, see Section 4.5.4.2: *Core Server Tape Volume Information Window* on page 271.

Drive Error Limit. This field is intended to be used in conjunction with the PVR Server **Retry Mount Time Limit**. If the number of consecutive mount errors which occur to any drive in this PVR equal or exceed this value, the drive is automatically locked by the PVL. The only mount errors that apply are those set through the **Retry Mount Time Limit** mechanism. The **Drive Error Count** field in the *PVL Drive Information* records the number of consecutive errors on a drive by drive basis. To turn off the automatic drive disable feature, set the **Drive Error Limit** to 0 or -1. Since this field is employed by the PVL, changing its value will not change drive disable behavior until the PVL is recycled.

Characteristics. Flags for the PVR:

- **Defer Dismounts.** If ON, the PVL will delay the dismounting of a tape cartridge until the drive is required by another job or until the **Dismount Delay** time limit is exceeded.
- **Support Shelf Tape.** If ON, the PVR and the PVL will support the removal of cartridges from the tape library using the shelf_tape utility.

ACSLS Packet Version. The packet version used by STK's ACSLS software.

Alternate SSI Port. A site running two SSI (Server System Interface) services on the same platform will specify the non-default port number of the second service via this field.

5.1.1.1.1. STK PVR Additional Information

When configuring HPSS to manage an STK Robot, the Drive Address configuration entries correspond to the ACS, Unit, Panel, and Drive number used by ACSLS to identify drives. For example, the first drive in a typical STK Robot configuration has the Drive Address (0,0,10,0).

HPSS can share an STK Robot with other tape management systems. If a robot is shared, care must be taken to make sure that a drive is not used by any other tape management system while that drive is configured as unlocked in HPSS. This is important because HPSS can be configured to periodically poll all of its unlocked drives even if they are not currently mounted or in use by HPSS. If a drive is being used by another tape management system, it must be configured as locked in HPSS.

HPSS will use any Cartridge Access Port (CAP) in the STK Robot that has a priority greater than zero. When it needs a CAP, HPSS will pick the highest priority CAP that is currently available. At least one CAP must be assigned a non-zero priority. See the STK Automated Cartridge System Library Software (ACSLS) System Administrator's Guide for procedures to set CAP priority.

ACSLS Packet Version

See the STK Automated Cartridge System Library Software (ACSLS) System Administrator's Guide for details. The environment variable **ACSAPI_PACKET_VERSION** will override the value entered in the **ACSLS Packet Version** field of the *STK PVR Specific Configuration Window*. If neither are set, a default value of 4 is used since ACSLS 6.x generally uses packet version 4.

Starting the STK Client Processes

In order for the HPSS PVR to communicate with the STK robot, it is required that STK client side processes be running on the node where the HPSS PVR is executing. These client side processes are the Server System Interface (SSI) and the Toolkit event logger. These binaries and associated script files are distributed with the HPSS, but are maintained by Sun Microsystems Inc.

The AIX binaries and script files for starting the STK client side processes on an AIX platform are located in the **$HPSS_PATH/stk/bin** directory. Documentation files describing the files in the bin directory are located in the **$HPSS_PATH/stk/doc** directory. Refer to these doc files for additional information. Similarly, Linux binaries and script files for starting the STK client side processes on a Linux platform are located in the **$HPSS_PATH/stk_linux/bin** directory; and associated documentation is located in the **$HPSS_PATH/stk_linux/doc** directory.

The **t_startit.sh** file must be executed to start the STK client processes on those nodes where the HPSS PVR is executing. The script will prompt for the following options:

1. Multi-Host or Single-Host - specify m for multiple host.
2. Server side or Client side - Specify c for client side.
3. Remote Host Name - specify the name of the host where the ACSLS software is running.
4. Real ACSLS server or simulated test – specify r for real ACSLS server.
5. Remote Host Version - specify the remote host version which is listed for the ACSLS Release Number. This will usually be 4:

ACSLS Release Number	Remote Host Version Number
5.x, 6.x	4

6. Whether processes created should be listed (Y or N) - specify either y or n.
7. Whether t_cdriver should be started - specify n.

To terminate the STK client processes, the script **t_killit.sh** may be executed.

Sample output from **t_startit.sh** follows:

```
            ***** Welcome to t_startit.sh, Version 2.3 *****
    This is the automated tester and startit script for the TOOLKIT.
    Simply answer the questions which follow.
    Executables live in: /opt/hpss/stk/bin
```

```
Would you like Multi-Host or Single-Host testing?
    Enter one of the following followed by ENTER:
    M Multi-host testing
    S Single-host testing
    X eXit this script
Enter choice: m
Would you like to define the server side or client side
for Multi-Host testing?
    Enter one of the following followed by ENTER:
    S Server side
    C Client side
Enter choice: c
The Remote Host Name is the name of the server which
has the ACSLS software (or simulator) running on it.
Enter Remote Host Name (CSI_HOSTNAME): jeep
Is that host running a real ACSLS (or LibStation) server,
or a Simulated Test server (t_acslm)?
    Enter one  of the following followed by ENTER:
    R   Real ACSLS or LibStation Server
    S   Simulated Test Server
Enter choice: r
  The Remote Host Version number is the ACSLS Packet
  Version level which the server is expecting.
  Here are the valid choices:
  ACSLS Release Number     Remote Host Version Number

          3.x                         2
          4.x                         3
        5.x, 6.x                      4
Enter Remote Host Version (ACSAPI_PACKET_VERSION): 4
Starting /opt/hpss/stk/bin/mini_el...
Attempting startup of /opt/hpss/bin/mini_el ...
Starting /opt/hpss/bin/ssi...
Attempting startup of PARENT for /opt/hpss/bin/ssi...
SIGHUP received Parent Process ID is: 17290
Attempting startup of /opt/hpss/bin/ssi...
SIGHUP received Parent Process #17290 EXITING NORMALLY
Initialization Done.
Do you want to see the processes created? (Y or N): y
17288 p4 S 0:00 /opt/hpss/stk/bin/mini_el
17292 p4 S 0:00 /opt/hpss/stk/bin/ssi 17290 50004 23
17295 p4 S 0:00 grep /opt/hpss/stk/bin
Do you want to start up t_cdriver? (Y or N): n
```

5.1.1. Deleting a Server Configuration

A server's configuration can be removed when the server is no longer needed. However, care should be taken to ensure that all objects created by the server are deleted properly and that all references to the server are removed from the HPSS configuration.

> *The steps described in this section are general guidelines. Specific procedures should be worked out with the aid of HPSS technical support so that the details of the system's configuration can be considered.*

A server's configuration should be removed only when it is no longer needed. To modify a server's configuration, update the existing configuration instead of deleting the configuration and creating a new one. This will reduce the possibility of introducing configuration errors into HPSS.

There must be at least one Core Server, PVL, and Mover configuration entry in order for HPSS to be operational.

It is very important that the server's configuration entries be deleted in the order shown below.

To delete a server, perform the following steps in the order shown:

1. Delete all objects created by the server.
   - If deleting a Core Server configuration, realize that this should only be done when deleting the full subsystem. For detailed information on deleting a subsystem, see Section 4.2.5: *Deleting a Storage Subsystem* on page 81.
   - If deleting a PVL configuration, all imported cartridges must first be exported.
   - If deleting a PVR configuration, all injected cartridges must either be exported or moved to another PVR.
   - If deleting a Gatekeeper, remove all references to that Gatekeeper from every Subsystem Configuration.
1. Remove references to the server from other configurations.

   Do not delete Root Core Server configuration.

   - If deleting a PVR configuration, update the **PVR** field in the appropriate Device and Drive configuration using the *Devices and Drives* window (Section 7.1.1 on page 202).
   - If deleting a Mover configuration, update the Mover field in the appropriate Device and Drive configuration using the *Devices and Drives* window (Section 7.1.1 on page 202).
1. After the affected configurations are modified, reinitialize (if supported) or recycle the appropriate HPSS servers.
2. Ensure that the server is not running.
3. Delete UNIX files created/used by the server, if no longer needed.
   - If deleting the MPS, delete the MPS reports.
   - If deleting the Startup Daemon, delete all server lock files in the /var/hpss/tmp directory. The names of these files always begin with hpssd.
   - If deleting the Mover, delete Mover's socket files in /var/hpss/tmp directory.
1. Delete the server configuration by opening the server list, highlighting the target server, and

pressing the **Delete** button.

5.1. Monitoring Server Information

A server that is running and connected to SSM will allow the SSM user to view and update its information. This section describes the server execution statuses and configuration information.

A typical HPSS server allows the SSM users to control its execution and monitor its server related data through the *Basic Server Information* window and the server specific information windows. These windows are described in the following subsections.

5.1.1. Basic Server Information

This window allows you to view the basic information associated with an HPSS server. The information is acquired from the server itself. While this window remains open, it will be automatically updated when the status of the server changes.

The *Basic Server Information* window is not be available for viewing if the server's operational state is Broken, Not Enabled, or Unknown, or the System Manager is unable to contact the server.

Field Descriptions

Server ID. The UUID for the server.

Server Name. The descriptive name of the server.

Operational State. The Operational State of the server. This state indicates the overall health of the server. Possible values are:

- Enabled - The server is running normally.
- Disabled - The server is not running because it has been shut down by SSM. You will not normally see this state on this window since the server must be running to provide the state information.
- Suspect - The server has seen an error that may indicate a problem is developing.
- Minor - The server has seen a problem which does not seriously impact HPSS operation.
- Major - The server has seen a problem which degrades HPSS operation.
- Broken - The server has shut itself down due to a serious error.

Usage State. The Usage State of the server. Possible values are:

- Active - The server is working normally.
- Idle - The server is running, but has no work to perform. Most servers do not update Usage State dynamically, so it is unlikely you will see this value reported.

- Busy - The server is busy performing its function. Most servers do not update Usage State dynamically, so it is unlikely you will see this value reported.
- Unknown - The server has not reported a recognized Usage State.

Administrative State. The Administrative State of the server. The possible states are:

- Shut Down - The server shut itself down in an orderly way.
- Force Halt - The server accepted a Force Halt command and terminated immediately.
- Reinitialize - The server reinitialized itself. Not all HPSS servers support being reinitialized.
- Mark Repaired - The server cleared its error states.
- Unlocked - Displayed during normal server operation.

Execution State. The Execution State of the server. Possible values are: Active, Initializing, Restarting, Terminating, Inactive, Suspending, Suspended, and Resuming.

Service Status. The server's service status. Possible values are: Normal, Warning, Minor Problem, Major Problem, and Critical.

Security Status. The server's security status. Possible values are: Normal, Warning, Minor Problem, Major Problem, and Critical.

Software Status. The server's software status. Possible values are: Normal, Warning, Minor Problem, Major Problem, and Critical.

Hardware Status. The server's hardware status. Possible values are: Normal, Warning, Minor Problem, Major Problem, and Critical.

Communication Status. The server's communication status. Possible values are: Normal, Warning, Minor Problem, Major Problem, and Critical.

Using the Basic Server Information Window

Most HPSS servers allow the SSM user to view the server's states and statuses.

Under *normal* conditions, the server states and statuses reported on the *Server Information* window are as follows:

Operational State: **Enabled**

Usage State: **Active**

Administrative State: **Unlocked**

Execution State: **Active**

Service Status: **Normal**

Security Status: **Normal**

Software Status: **Normal**

Hardware Status: **Normal**

Communication Status: **Normal**

However, when the server is experiencing errors or encountering abnormal conditions, it will change the appropriate states and statuses to error values, notify SSM of the changes, and issue an alarm to SSM. Refer to Section 9.6.2: *Alarm/Event Information* on page 303 for more information.

The Startup Daemon and the System Manager do not have their own Basic Server Information windows.

5.1.1. Specific Server Information

Using the Server Specific Information Windows

The majority of the HPSS servers also allow the SSM user to view and change their server specific data through the SSM windows. A typical server allows authorized users to change the value of the fields and use them immediately in the current execution.

There are server specific information windows for the following servers:

- Core Server
- Gatekeeper
- Location Server
- Migration/Purge Server
- Mover
- Physical Volume Library
- Physical Volume Repository

Any changes made on a server's information window will be effective only during the server's current execution. To make the changes permanent, make the identical changes on the server's configuration window and restart the server (see Section 5.1:Server Configuration on page 87 and below).

If the server's operational state is Broken or Unknown, the *Specific Server Information* window will not be available for viewing.

The server specific information windows are described in detail below.

5.1.1.1. Core Server Information Window

Field Descriptions

Core Server. The descriptive name of the Core Server.

Global Database Name. The name of the global database for the HPSS system.

Subsystem Database Name. The name of the database which contains the subsystem tables used by this Core Server.

Schema Name. The name of the database schema.

Root Fileset Name. The name of the root fileset used by the Core Server.

Root Fileset ID. The fileset id for the root fileset used by the Core Server.

Maximum Open Bitfiles. The maximum number of bitfiles that can be open simultaneously.

Maximum Active I/O Reqs. The maximum number of simultaneous I/O requests allowed.

Maximum Active Copy Reqs. The maximum number of simultaneous copy requests allowed.

COS Change Stream Count. Indicates the number of background COS change threads to run in the Core Server. The default is 1.

COS Change Retry Limit. Limits the number of times that the Core Server will retry a failed COS change operation. When this limit is reached, the attempt to change the COS of the file is terminated.

Tape Dismount Delay (seconds). The amount of time, in seconds, a mounted tape volume will remain idle before being dismounted by the Core Server. Larger values may reduce undesirable tape dismount/remount events, at the expense of lower tape drive utilization.

Tape Handoff Delay (seconds). The amount of time, in seconds, a mounted tape will be held in a client's session before become eligible to be handed off to another client that wishes to use the tape.

PVL Max Connection Wait (seconds). The amount of time, in seconds, the Core Server will wait to connect to the PVL before declaring an error in pending PVL jobs.

Fragment Trim Limit (clusters). Fragment Trim Limit sets the lower limit of the number of clusters that will be trimmed from a disk segment extent, when the extent turns out to be longer than the needed length.

Fragment Smallest Block. Fragment Smallest Block sets the boundary used to determine where to remove excess clusters from allocations that have excess space. The smallest block returned to the free space map from the excess space at the end of an extent will be this size or larger.

Root User ID. The numerical id for the root user used by the Core Server.

Root is Superuser. If this flag is ON, clients using the Root User ID will have superuser powers and permissions.

Subsystem Statistics:

- **Stages.** The number of stage requests processed in the Core Server since startup or last reset of the statistics.
- **Migrations**. The number of migrate requests processed in the Core Server since startup or last reset of the statistics.
- **Purges.** The number of purge requests processed in the Core Server since startup or last reset of the statistics.

- **File Deletes.** The number of bitfile delete requests processed in the Core Server since startup or last reset of the statistics.
- **Last Reset Time.** The last time the subsystem statistics were reset. If this value is 0, the statistics have not been reset since server startup.

Name Space Statistics:

- **Files.** The number of files managed by this Core Server.
- **Directories.** The number of directories managed by this Core Server.
- **Symbolic Links.** The number of symbolic links managed by this Core Server.
- **Hard Links.** The number of hard links managed by this Core Server.
- **Junctions.** The number of junctions managed by this Core Server.
- **Filesets.** The number of filesets managed by this Core Server.
- **Total Name Space Objects.** The total number of namespace records in the database. Note that this total may not match the total achieved by summing the number of the above objects. Some objects, such as hard links, require more than one record.

Disk/Tape Statistics:

- **Total Disk Virtual Volumes.** The number of Disk Virtual Volumes managed by this Core Server.
- **Total Tape Virtual Volumes.** The number of Tape Virtual Volumes managed by this Core Server.
- **Tape Aggregates.** The number of tape aggregates managed by this Core Server. A tape aggregate is a single tape storage segment that contains more than one file.
- **Files in Tape Aggregates.** The number of files in the tape aggregates managed by this Core Server.
- **Bytes in Tape Aggregates.** The number of bytes stored in the tape aggregates managed by this Core Server.
- **Active Disk Bytes.** This is the sum of the bytes in the storage segments assigned to files, including slack space. Slack space is the disk space assigned to disk storage segments that is not occupied with file data. The sum of the lengths of files on disk will be less than the Active Disk Bytes because of the presence of slack space.
- **Free Disk Bytes**. This is the sum of the bytes not assigned to files. More precisely, it is the sum of the disk volume bytes not assigned to disk storage segments.
- **Active Tape Bytes.** This is the sum of the bytes assigned to files on tapes. It does not include deleted file space that has not been reclaimed.

- **Free Tape Bytes**. This is an estimate based on the sum of the estimated sizes of the partially written and unwritten tape volumes. It is not, and cannot be, an accurate value as the amount of data that can be written on tapes varies with individual tape volumes and data compression levels.

Options:

- **Can change UID to self if has Control Perm.** If this flag is ON, any user having Control permission to an object can change the UID of that object to their own (but not any other) UID. This field can be changed only from the *Global Configuration* window. It is displayed here for informational purposes only.
- **Can change UID if has Delete Perm on Security ACL.** If this flag is ON, any principal found on the Core Server Security ACL having Delete (and only Delete) permission may change the UID of an object they own to any other UID. This field can be changed only from the *Global Configuration* window. It is displayed here for informational purposes only.
- **Object names can contain unprintable characters.** If this flag is ON, no check will be made to ensure that new object and fileset names contain only printable characters. If this flag is OFF, new object and fileset names must be composed of only printable characters and if any unprintable characters are detected in these new names, an error will be returned. Printable characters are defined as all characters in the range from 0x20 to 0x7e (inclusive) in the 7 bit ASCII character set. This field can be changed only from the *Global Configuration* window. It is displayed here for informational purposes only.
- **COS Copy to Disk.** If ON, all copy operations associated with COS changes should be directed to disk if the hierarchy of the target COS has a level 0 disk storage class. This flag may be changed only from the Specific tab of the *Core Server Configuration* window. It is displayed here for informational purposes only.
- **Account Flag Switch.** If this flag is ON, it indicates to use the second accounting bit in the bitfile metadata table to indicate that the bitfile has valid accounting metadata. If the flag is OFF, it indicates to use the first bit. The default is OFF. This flag setting is changeable only via special procedures. It is provided to allow a site to recover or rebuild its accounting metadata if this becomes necessary. Special procedures are required for this. Contact the HPSS support staff to do this.

5.1.1.1. Gatekeeper Information Window

Field Descriptions

Server Name. The descriptive name of the Gatekeeper.

Default Wait Time. The number of seconds the client must wait before retrying a request. This value must be greater than zero and is used if the Gatekeeping Site Interface returns a wait time of zero for the create, open, or stage request being retried. Changing the value of this field will cause the Gatekeeper to use the new value until the next restart at which point it will then go back to using the value defined in the *Gatekeeper Configuration* window. Refer to Section 5.1.2 *Gatekeeper Specific Configuration* on page 98.

Site Policy Pathname (UNIX). This field can only be set from the *Gatekeeper Configuration* window. Refer to Section 5.1.2 *Gatekeeper Specific Configuration* on page 98.

Administrative Activity Statistics:

These fields are internal volatile statistics for the Gatekeeper. Each row describes a different API. There are three columns: Calls, Errors and Retries. Each of these columns represent a count of the number of calls, errors, and retries since the Statistics were (re)set. The Calls column is the number of times the API was called. The Errors column is the number of times the API call resulted in an error being returned. The Retries column is the number of times that the API call resulted in the HPSS_ERETRY error being returned to the Client API. The Retries column only applies to API requests that may return the HPSS_ERETRY status. A retry is not counted as an error.

The Statistics are (re)set to zero whenever the Gatekeeper is recycled or the **Reset** button is clicked.

- **Get Basic Server Info**. Statistics from the gk_admin_ServerGetAttrs API.
- **Set Basic Server Info**. Statistics from the gk_admin_ServerSetAttrs API. This API is called by the SSM System Manager when the Gatekeeper's Basic Server Information window is opened.
- **Get Gatekeeper Server Info**. Statistics from the gk_admin_GKGetAttrs API.
- **Set Gatekeeper Server Info**. Statistics from the gk_admin_GKSetAttrs API. This API is called by the SSM System Manager when the Gatekeeper Information window is opened.
- **Creates**. Statistics from the gk_Create API. This API is called by the Core Server when the Gatekeeper is monitoring Create Requests.
- **Auth Caller Creates**. Statistics from authorized caller (e.g. MPS) calls to the gk_Create API.
- **Creates Completed**. Statistics from the gk_CreateComplete API. This API is called by the Core Server when the Gatekeeper is monitoring Create Requests and the create completes.
- **Opens**. Statistics from the gk_Open API. This API is called by the Core Server when the Gatekeeper is monitoring Open Requests.
- **Auth Caller Opens**. Statistics from authorized caller (e.g. MPS) calls to the gk_Open API.
- **Closes**. Statistics from the gk_Close API. This API is called by the Core Server when the Gatekeeper is monitoring Open Requests and the file is closed.
- **Stages**. Statistics from the gk_Stage API. This API is called by the Core Server when the Gatekeeper is monitoring Stage Requests.
- **Auth Caller Stages**. Statistics from authorized caller (e.g. MPS) calls to the gk_Stage API.
- **Stages Completed**. Statistics from the gk_StageComplete API. This API is called by the Core Server when the Gatekeeper is monitoring Stage Requests and the stage completes.
- **Client Clean Ups**. Statistics from the gk_CleanUpClient API. This API is called by the Core

Server when the Gatekeeper is monitoring Requests and a client disconnects.

- **Get Monitor Types**. Statistics from the gk_GetMonitorTypes API. This API is called by the Core Server to figure out what types of Requests being monitored by the Gatekeeper.
- **Pass Thrus**. Statistics from the gk_PassThru API.
- **Queries**. Statistics from the gk_Query API.
- **Read Site Policies**. Statistics from the gk_ReadSitePolicy API.
- **Last Reset Time**. The time stamp when the Statistics were last (re)set to zero.

Associated Button Descriptions

Reset. Reset the Administrative Activity Statistics to 0 and the **Last Reset Time** to the current date and time.

Read Site Policy. This button is used to inform the GatekeepingSite Interface (gk_site_ReadSitePolicy) of a policy change to the Site Policy Pathname file. Note: a "policy change" cannot include a change in the types of requests being monitored. When changing the types of requests being monitored (authorized caller, create, open, and stage), the Gatekeeper must be recycled so that the Core Server can learn which request types need to include Gatekeeping services.

5.1.1.1. Location Server Information Window

The *Location Server Information* window displays statistics for the server's request processing and background updating functions.

The maximum number of requests a single Location Server can process in one minute varies depending on the machine load caused by other processes as well as the CPU performance. Generally the Location Server will report warning messages when it becomes loaded.

If timeouts occur, consider increasing the **Location Map Timeout** field on the *Location Policy* window.

Field Descriptions

Requests Per Minute. The number of client requests received during the last minute.

Request Errors. The number of errors returned to client requests.

Location Map Requests. The number of client requests received for local server location information (maps).

Location Map Updates. The number of background location map updates that have occurred.

Location Map Update Timeouts. The number of timeouts that have occurred while waiting for location map information.

Minimum Location Map Update Time. The shortest time, in seconds, needed for a location map update.

Average Location Map Update Time. The average time, in seconds, needed for a location map update.

Maximum Location Map Update Time. The longest time, in seconds, needed for a location map update.

Associated Button Descriptions

Reset. Reset the Statistics to 0**.**

Related Information

HPSS Error Manual: Chapter 1, Section 1.2.10: *Location Server Problems.*

5.1.1.2. Migration/Purge Server Information Window

This window allows you to view and update the type-specific information associated with a Migration/Purge Server (MPS).

Any changes made to fields on this window are sent directly to the MPS and are effective immediately.

Field Descriptions

Server Name. The descriptive name of the MPS.

Storage Class Update Interval. The MPS periodically queries each Core Server to get up-to-date statistics on each storage class that the Core Server manages. This field is the interval in seconds between successive queries.

Maximum Volumes for Whole File Migration. The limit for the number of tape volumes mounted simultaneously for each storage class using the Whole File or Whole File With Purge migration options. Migration of other storage classes, end user stage requests, and many other factors should be considered when deciding on this value.

Core Server API Failures Allowed. This field sets a limit on the number of errors returned from the Core Server API. This applies to both disk and tape migration as well as disk purge. If this limit is reached during a disk migration run, migration skips to the next hierarchy to which that disk storage class belongs. Once the run has attempted to migrate all such hierarchies the run will end. If this limit is reached during a disk purge or tape migration run, the migration or purge run aborts.

5.1.1.3. Mover Information Window

This window allows you to view and update the type-specific information associated with a Mover. Any changes made to fields on this window are sent directly to the Mover after the appropriate button is pressed and are effective immediately.

Field Descriptions

Server Name. The descriptive name of the Mover.

Number of Request Tasks. The number of Mover request-processing tasks that currently exist.

Number of Active Requests. The number of active requests that are being handled by the Mover.

Time of Last Statistics Reset. The time and date when the Mover statistics were last reset.

Buffer Size. The size, in bytes, of each buffer used by the Mover when transferring data. Each Mover task uses two buffers of this size to perform double buffering during I/O operations. See the *Mover Configuration* window **Buffer Size** field described in Section 5.1.3: *Mover Specific Configuration* on page 102 for more detailed information.

Number of Requests Processed. The number of requests that the Mover has processed since the last time this field was cleared. Click on the **Reset** button to the right of the field to reset the value to zero.

Number of Request Errors. The number of requests that have returned errors since the last time this field was cleared. Click on the **Reset** button to the right of the field to reset the value to zero.

Number of Data Transfers. The number of data transfers that the Mover has handled since the last time this field was cleared. Click on the **Reset** button to the right of the field to reset the value to zero.

Number of Bytes Moved. The number of bytes of data that the Mover has transferred since the last time this field was cleared. Click on the **Reset** button to the right of the field to reset the value to zero.

Total Move Time. The total time, in days, hours, minutes, and seconds, that the Mover tasks have spent transferring data since the last time this field was cleared. Click on the **Reset** button to the right of the field to reset the value to zero.

Operational Notes

1. The Mover statistics (Number of Requests Processed, Number of Request Errors, Number of Data Transfers, Number of Bytes Moved and Total Move Time) may all be reset to zero. If any of these fields are reset the Time of **Last Statistics Reset** field will also be updated with the current time. The statistics fields are all initialized to zero when the Mover is started.
2. The Total Move Time statistic should not be used to determine Mover throughput rates if the Mover handles striped media, since this is a total for all Mover tasks involved in the transfer. This total will most likely be considerably greater than the elapsed time necessary to transfer the data.

Related Information

HPSS Error Manual, Chapter 1, Section 1.2.7: *Mover Problems.*

5.1.1.1. Physical Volume Library (PVL) Information Window

This window allows you to view the type-specific information associated with a PVL.

Field Descriptions

Server Name. The descriptive name of the PVL.

Total Volumes. The total number of volumes that have been imported into the PVL.

Total Repositories. The total number of PVRs in the *Servers* list window.

Total Drives. The total number of drives controlled by this PVL.

5.1.1.2. Physical Volume Repository (PVR) Information Windows

These windows allow you to view/update the type-specific information associated with a PVR.

Any changes made to fields on these windows are sent directly to the PVR and are effective immediately.

5.1.1.2.1. Operator PVR Information Window

Field Descriptions

Server Name. The descriptive name of the PVR.

Total Cartridges. The number of cartridges currently being managed by the PVR.

Cartridge Capacity. The total number of cartridge slots in the library dedicated to this HPSS PVR. This may or may not be the total cartridge capacity of the library; a site might use part of the library for some other HPSS PVR or for some non-HPSS application. The PVR uses the **Cartridge Capacity** field and the **Cartridge Alarm Threshold** field to determine when to send an alarm that the total cartridge threshold has been exceeded.

Cartridge Alarm Threshold. The percentage of the **Cartridge Capacity** at which the PVR will send an alarm.

Dismount Delay. When **Defer Dismounts** is checked, this value is used by the PVL to determine the number of minutes that dismounts are delayed after the last data access.

Shelf Tape Check-In Alarm. The PVR will periodically log alarm messages when a requested shelf tape has not been checked-in. This field specifies the number of minutes between alarms. This field is only active if the **Support Shelf Tape** checkbox is selected. The alarm value must be 2 or greater.

Shelf Tape Check-In Retry. The number of seconds the PVR will wait before asking the robot if a requested shelf tape has been checked-in. The PVR will continue checking at this interval until the tape is checked-in. This field applies only if the **Support Shelf Tape** checkbox is selected. The retry value must be 30 or greater.

Characteristics. Flags for the PVR:

- **Defer Dismounts.** If ON, the PVL will delay the dismounting of a tape cartridge until the drive is required by another job or until the **Dismount Delay** time limit is exceeded.
- **Support Shelf Tape.** If ON, the PVR and the PVL will support the removal of cartridges from the tape library using the shelf_tape utility.

5.1.1.2.1. 3494 PVR Information Window

Field Descriptions

Server Name. The descriptive name of the PVR.

Total Cartridges. The number of cartridges currently being managed by the PVR.

Cartridge Capacity. The total number of cartridge slots in the library dedicated to this HPSS PVR. This may or may not be the total cartridge capacity of the library; a site might use part of the library for some other HPSS PVR or for some non-HPSS application. The PVR uses the **Cartridge Capacity** field and the **Cartridge Alarm Threshold** field to determine when to send an alarm that the total cartridge threshold has been exceeded.

Cartridge Alarm Threshold. The percentage of the **Cartridge Capacity** at which the PVR will send an alarm.

Same Job on Controller, **Other Job on Controller**, & **Distance To Drive**. These values are used by the PVR when selecting a drive for a tape mount operation. The three values are essentially weights that are used to compute an overall score for each possible drive. After the score has been calculated, the drive with the lowest score is selected for the mount. If two or more drives tie for the lowest score, one drive is selected at random. The score is calculated as follows:

Score =

Weight 1 * Cartridges from this job mounted on this drive's controller +

Weight 2 * Cartridges from other jobs mounted on this drive's controller +

Weight 3 * Units of distance from the cartridge to the drive

This method has the effect of distributing a striped tape mount across as many controllers as possible for the best performance. It also will try to pick controllers that are currently driving a minimum number of tapes. So, in an environment with many tape drives per controller, the above algorithm will minimize the load on any one controller.

The **Distance To Drive** helps minimize mount times by mounting the tape in a physically close drive. For IBM robots, the time it takes to move a cartridge to a more distant drive is not significant, so the default value here is fairly low. The unit of distance for a 3494 library is one frame. All other things being equal, the tape will be mounted in the closest drive. However, if the closest drive is attached to a heavily used controller, then a more distant drive will be selected.

Retry Mount Time Limit. The default value for this field is -1. When the default value (-1) is used, if an error is encountered during a PVR mount operation, the mount will pend and be retried every 5 minutes. Setting a value in this field will change the mount behavior to periodically retry the mount until the specified time limit is exceeded. Once exceeded, an error is generated and the mount request is canceled. If the mount request would have resulted in a write operation, the error returned will cause the Core Server to set the **VV Condition** of the associated tape volume to DOWN. Once in DOWN state, the volume will no longer be available for read or write operations. For further information about the Core Server VV Condition, see Section 4.5.4.2: *Core Server Tape Volume Information Window* on page 271.

Drive Error Limit. This field is used in conjunction with the PVR Server **Retry Mount Time Limit** field. If the number of consecutive mount errors which occur to any drive in this PVR equal or exceed this value, the drive is automatically locked by the PVL. The only mount errors that apply are those set through the **Retry Mount Time Limit** mechanism. The **Drive Error Count** field in the PVL Drive Information records the number of consecutive errors on a drive by drive basis. To turn off the automatic drive disable feature, set the **Drive Error Limit** to 0 or -1. Changing this value will not change drive disable behavior until the PVL is recycled.

Command Device. The name of the device that the PVR can use to send commands to the robot. For AIX systems, this is generally **/dev/lmcp0**. For Linux systems, use the symbolic library name defined in **/etc/ibmatl.conf**. The environment variable **HPSS_3494_COMMAND_DEVICE** will override the value entered in this field. Only use the environment variable when using only one 3494.

Async Device. The name of the device that the PVR can use to receive replies from the 3494 robot. For AIX systems, this is generally **/dev/lmcp1**. For Linux systems, use the symbolic library name defined in **/etc/ibmatl.conf**. For TTY and LAN-attached robots, the **Async Device** may be the same as the **Command Device**. The environment variable **HPSS_3494_ASYNC _DEVICE** will override the value entered in this field. Only use the environment variable when using only one 3494.

Dismount Delay. When **Defer Dismounts** is checked, this value is used by the PVL to determine the number of minutes that dismounts are delayed after the last data access.

Shelf Tape Check-In Retry. The number of seconds the PVR will wait before asking the robot if a requested shelf tape has been checked-in. The PVR will continue checking at this interval until the tape is checked-in. This field applies only if the **Support Shelf Tape** checkbox is selected. The retry value must be 30 or greater.

Shelf Tape Check-In Alarm. The PVR will periodically log alarm messages when a requested shelf tape has not been checked-in. This field specifies the number of minutes between alarms. This field is only active if the **Support Shelf Tape** checkbox is selected. The alarm value must be 2 or greater.

Characteristics. Flags for the PVR:

- **Defer Dismounts.** If ON, the PVL will delay the dismounting of a tape cartridge until the drive is required by another job or until the **Dismount Delay** time limit is exceeded.
- **Support Shelf Tape.** If ON, the PVR and the PVL will support the removal of cartridges from the tape library using the shelf_tape utility.

5.1.1.2.1. AML PVR Information Window

Field Descriptions

Server Name. The descriptive name of the PVR.

Total Cartridges. The number of cartridges currently being managed by the PVR.

Cartridge Capacity. The total number of cartridge slots in the library dedicated to this HPSS PVR. This may or may not be the total cartridge capacity of the library; a site might use part of the library for some other HPSS PVR or for some non-HPSS application. The PVR uses the **Cartridge Capacity** field and the **Cartridge Alarm Threshold** field to determine when to send an alarm that the total cartridge threshold has been exceeded.

Cartridge Alarm Threshold. The percentage of the **Cartridge Capacity** at which the PVR will send an alarm.

Same Job on Controller, **Other Job on Controller**, & **Distance To Drive**. These values are used by the PVR when selecting a drive for a tape mount operation. The three values are essentially weights that are used to compute an overall score for each possible drive. After the score has been calculated, the drive with the lowest score is selected for the mount. If two or more drives tie for the lowest score, one drive is selected at random. The score is calculated as follows:

Score =

> Weight 1 * Cartridges from this job mounted on this drive's controller +
>
> Weight 2 * Cartridges from other jobs mounted on this drive's controller +
>
> Weight 3 * Units of distance from the cartridge to the drive

This method has the effect of distributing a striped tape mount across as many controllers as possible for the best performance. It also will try to pick controllers that are currently driving a minimum number of tapes. So, in an environment with many tape drives per controller, the above algorithm will minimize the load on any one controller.

The **Distance To Drive** helps minimize mount times by mounting the tape in a physically close drive. All other things being equal, the tape will be mounted in the closest drive. However, if the closest drive is attached to a heavily used controller, then a more distant drive will be selected.

Retry Mount Time Limit. The default value for this field is -1. When the default value (-1) is used, if an error is encountered during a PVR mount operation, the mount will pend and be retried every 5 minutes. Setting a value in this field will change the mount behavior to periodically retry the mount until the specified time limit is exceeded. Once exceeded, an error is generated and the mount request is canceled. If the mount request would have resulted in a write operation, the error returned will cause the Core Server to set the **VV Condition** of the associated tape volume to DOWN. Once in DOWN state, the volume will no longer be available for read or write operations. For further information about the Core Server VV Condition, see Section 4.5.4.2: *Core Server Tape Volume Information Window* on page 271.

Drive Error Limit. This field is used in conjunction with the PVR Server **Retry Mount Time Limit**. If the number of consecutive mount errors which occur to any drive in this PVR equal or exceed this value, the drive is automatically locked by the PVL. The only mount errors that apply are those set through the **Retry Mount Time Limit** mechanism. The **Drive Error Count** field in the PVL Drive Information records the number of consecutive errors on a drive by drive basis. To turn off the automatic drive disable feature, set the **Drive Error Limit** to 0 or -1. Changing this value will not change drive disable behavior until the PVL is recycled.

Client Name . The name of the client requesting authorization from the Distributed Automated Media Library Server.

Server Name. TCP/IP host name or IP address of the AML OS/2-PC DAS server.

Dismount Delay. When **Defer Dismounts** is checked, this value is used by the PVL to determine the number of minutes that dismounts are delayed after the last data access.

Shelf Tape Check-In Retry. The number of seconds the PVR will wait before asking the robot if a requested shelf tape has been checked-in. The PVR will continue checking at this interval until the tape is checked-in. This field applies only if the **Support Shelf Tape** checkbox is selected. The retry value must be 30 or greater.

Shelf Tape Check-In Alarm. The PVR will periodically log alarm messages when a requested shelf tape has not been checked-in. This field specifies the number of minutes between alarms. This field is only active if the **Support Shelf Tape** checkbox is selected. The alarm value must be 2 or greater.

Characteristics. Flags for the PVR:

- **Defer Dismounts.** If ON, the PVL will delay the dismounting of a tape cartridge until the drive is required by another job or until the **Dismount Delay** time limit is exceeded.
- **Support Shelf Tape.** If ON, the PVR and the PVL will support the removal of cartridges from the tape library using the shelf_tape utility.

5.1.1.2.1. LTO PVR Information Window

Field Descriptions

Server Name. The descriptive name of the PVR.

Total Cartridges. The number of cartridges currently being managed by the PVR.

Cartridge Capacity. The total number of cartridge slots in the library dedicated to this HPSS PVR. This may or may not be the total cartridge capacity of the library; a site might use part of the library for some other HPSS PVR or for some non-HPSS application. The PVR uses the **Cartridge Capacity** field and the **Cartridge Alarm Threshold** field to determine when to send an alarm that the total cartridge threshold has been exceeded.

Cartridge Alarm Threshold. The percentage of the **Cartridge Capacity** at which the PVR will send an alarm.

Same Job on Controller, **Other Job on Controller**, & **Distance To Drive**. These values are used by the PVR when selecting a drive for a tape mount operation. The three values are essentially weights that are used to compute an overall score for each possible drive. After the score has been calculated, the drive with the lowest score is selected for the mount. If two or more drives tie for the lowest score, one drive is selected at random. The score is calculated as follows:

Score =

Weight 1 * Cartridges from this job mounted on this drive's controller +

Weight 2 * Cartridges from other jobs mounted on this drive's controller +

Weight 3 * Units of distance from the cartridge to the drive

This method has the effect of distributing a striped tape mount across as many controllers as possible for the best performance. It also will try to pick controllers that are currently driving a minimum number of tapes. So, in an environment with many tape drives per controller, the above algorithm will minimize the load on any one controller.

The **Distance To Drive** helps minimize mount times by mounting the tape in a physically close drive. For IBM robots, the time it takes to move a cartridge to a more distant drive is not significant, so the default value here is fairly low. All other things being equal, the tape will be mounted in the closest drive. However, if the closest drive is attached to a heavily used controller, then a more distant drive will be selected.

Retry Mount Time Limit. The default value for this field is -1. When the default value (-1) is used, if an error is encountered during a PVR mount operation, the mount will pend and be retried every 5 minutes. Setting a value in this field will change the mount behavior to periodically retry the mount until the specified time limit is exceeded. Once exceeded, an error is generated and the mount request is canceled. If the mount request would have resulted in a write operation, the error returned will cause the Core Server to set the **VV Condition** of the associated tape volume to DOWN. Once in DOWN state, the volume will no longer be available for read or write operations. For further information about the Core Server VV Condition, see Section 4.5.4.2: *Core Server Tape Volume Information Window* on page 271.

Drive Error Limit. This field is used in conjunction with the PVR Server **Retry Mount Time Limit**. If the number of consecutive mount errors which occur to any drive in this PVR equal or exceed this value, the drive is automatically locked by the PVL. The only mount errors that apply are those set through the **Retry Mount Time Limit** mechanism. The **Drive Error Count** field in the PVL Drive Information records the number of consecutive errors on a drive by drive basis. To turn off the automatic drive disable feature, set the **Drive Error Limit** to 0 or -1. Changing this value will not change drive disable behavior until the PVL is recycled.

Command Device. The name of the device that the PVR can use to send commands to the robot. For AIX systems, this is generally **/dev/smc0**. For Linux systems, use the symbolic library name defined in **/etc/ibmatl.conf**.

Dismount Delay. When **Defer Dismounts** is checked, this value is used by the PVL to determine the number of minutes that dismounts are delayed after the last data access.

Shelf Tape Check-In Retry. The number of seconds the PVR will wait before asking the robot if a requested shelf tape has been checked-in. The PVR will continue checking at this interval until the tape is checked-in. This field applies only if the **Support Shelf Tape** checkbox is selected. The retry value must be 30 or greater.

Shelf Tape Check-In Alarm. The PVR will periodically log alarm messages when a requested shelf tape has not been checked-in. This field specifies the number of minutes between alarms. This field is only active if the **Support Shelf Tape** checkbox is selected. The alarm value must be 2 or greater.

Characteristics. Flags for the PVR:

- **Defer Dismounts.** If ON, the PVL will delay the dismounting of a tape cartridge until the drive is required by another job or until the **Dismount Delay** time limit is exceeded.
- **Support Shelf Tape.** If ON, the PVR and the PVL will support the removal of cartridges from the tape library using the shelf_tape utility.

5.1.1.2.1. SCSI PVR Information Window

Field Descriptions

Server Name. The descriptive name of the PVR.

Total Cartridges. The number of cartridges currently being managed by the PVR.

Cartridge Capacity. The total number of cartridge slots in the library dedicated to this HPSS PVR. This may or may not be the total cartridge capacity of the library; a site might use part of the library for some other HPSS PVR or for some non-HPSS application. The PVR uses the **Cartridge Capacity** field and the **Cartridge Alarm Threshold** field to determine when to send an alarm that the total cartridge threshold has been exceeded.

Cartridge Alarm Threshold. The percentage of the **Cartridge Capacity** at which the PVR will send an alarm.

Same Job on Controller, **Other Job on Controller**, & **Distance To Drive**. These values are used by the PVR when selecting a drive for a tape mount operation. The three values are essentially weights that are used to compute an overall score for each possible drive. After the score has been calculated, the drive with the lowest score is selected for the mount. If two or more drives tie for the lowest score, one drive is selected at random. The score is calculated as follows:

Score =

Weight 1 * Cartridges from this job mounted on this drive's controller +

Weight 2 * Cartridges from other jobs mounted on this drive's controller +

Weight 3 * Units of distance from the cartridge to the drive

This method has the effect of distributing a striped tape mount across as many controllers as possible for the best performance. It also will try to pick controllers that are currently driving a minimum number of tapes. So, in an environment with many tape drives per controller, the above algorithm will minimize the load on any one controller.

The **Distance To Drive** helps minimize mount times by mounting the tape in a physically close drive. All other things being equal, the tape will be mounted in the closest drive. However, if the closest drive is attached to a heavily used controller, then a more distant drive will be selected.

Retry Mount Time Limit. The default value for this field is -1. When the default value (-1) is used, if an error is encountered during a PVR mount operation, the mount will pend and be retried every 5 minutes. Setting a value in this field will change the mount behavior to periodically retry the mount until the specified time limit is exceeded. Once exceeded, an error is generated and the mount request is canceled. If the mount request would have resulted in a write operation, the error returned will cause the Core Server to set the **VV Condition** of the associated tape volume to DOWN. Once in DOWN state, the volume will no longer be available for read or write operations. For further information about the Core Server VV Condition, see Section 4.5.4.2: *Core Server Tape Volume Information Window* on page 271.

Drive Error Limit. This field is used in conjunction with the PVR Server **Retry Mount Time Limit**. If the number of consecutive mount errors which occur to any drive in this PVR equal or exceed this value, the drive is automatically locked by the PVL. The only mount errors that apply are those set through the **Retry Mount Time Limit** mechanism. The **Drive Error Count** field in the PVL Drive Information records the number of consecutive errors on a drive by drive basis. To turn off the automatic drive disable feature, set the **Drive Error Limit** to 0 or -1. Changing its value will not change drive disable behavior until the PVL is recycled.

Serial Number. The serial number of the robot the SCSI PVR is controlling. Used to detect control paths to the robot.

Dismount Delay. When **Defer Dismounts** is checked, this value is used by the PVL to determine the number of minutes that dismounts are delayed after the last data access.

Shelf Tape Check-In Retry. The number of seconds the PVR will wait before asking the robot if a requested shelf tape has been checked-in. The PVR will continue checking at this interval until the tape is checked-in. This field applies only if the **Support Shelf Tape** checkbox is selected. The retry value must be 30 or greater.

Shelf Tape Check-In Alarm. The PVR will periodically log alarm messages when a requested shelf tape has not been checked-in. This field specifies the number of minutes between alarms. This field is only active if the **Support Shelf Tape** checkbox is selected. The alarm value must be 2 or greater.

Characteristics. Flags for the PVR:

- **Defer Dismounts.** If ON, the PVL will delay the dismounting of a tape cartridge until the drive is required by another job or until the **Dismount Delay** time limit is exceeded.
- **Support Shelf Tape.** If ON, the PVR and the PVL will support the removal of cartridges from the tape library using the shelf_tape utility.
- **Enforce Home Location.** If ON, the SCSI PVR will always try to dismount a mounted cart back to its home location. Otherwise, it will just use the first free slot. The scsi_home utility can be used to view and manipulate the home location values.

5.1.1.2.1. STK PVR Information Window

Field Descriptions

Server Name. The descriptive name of the PVR.

Total Cartridges. The number of cartridges currently being managed by the PVR.

Cartridge Capacity. The total number of cartridge slots in the library dedicated to this HPSS PVR. This may or may not be the total cartridge capacity of the library; a site might use part of the library for some other HPSS PVR or for some non-HPSS application. The PVR uses the **Cartridge Capacity** field and the **Cartridge Alarm Threshold** field to determine when to send an alarm that the total cartridge threshold has been exceeded.

Cartridge Alarm Threshold. The percentage of the **Cartridge Capacity** at which the PVR will send an alarm.

Same Job on Controller, **Other Job on Controller**, & **Distance To Drive**. These values are used by the PVR when selecting a drive for a tape mount operation. The three values are essentially weights that are used to compute an overall score for each possible drive. After the score has been calculated, the drive with the lowest score is selected for the mount. If two or more drives tie for the lowest score, one drive is selected at random. The score is calculated as follows:

Score =

Weight 1 * Cartridges from this job mounted on this drive's controller +

Weight 2 * Cartridges from other jobs mounted on this drive's controller +

Weight 3 * Units of distance from the cartridge to the drive

This method has the effect of distributing a striped tape mount across as many controllers as possible for the best performance. It also will try to pick controllers that are currently driving a minimum number of tapes. So, in an environment with many tape drives per controller, the above algorithm will minimize the load on any one controller.

The **Distance To Drive** helps minimize mount times by mounting the tape in a physically close drive. For STK drives, a unit of distance is from one Silo/LSM (Library Storage Module) to the next. This means that the tape must go through a pass-through port or elevator. This process is more time consuming. Therefore the default value for STK robots is a higher value which forces the tape to stay in the same Silo/LSM even if it means the drive selected is attached to a controller which is heavily used. All other things being equal, the tape will be mounted in the closest drive.

Retry Mount Time Limit. The default value for this field is -1. When the default value (-1) is used, if an error is encountered during a PVR mount operation, the mount will pend and be retried every 5 minutes. Setting a value in this field will change the mount behavior to periodically retry the mount until the specified time limit is exceeded. Once exceeded, an error is generated and the mount request is canceled. If the mount request would have resulted in a write operation, the error returned will cause the Core

Server to set the **VV Condition** of the associated tape volume to DOWN. Once in DOWN state, the volume will no longer be available for read or write operations. For further information about the Core Server VV Condition, see Section 4.5.4.2: *Core Server Tape Volume Information Window* on page 271.

Drive Error Limit. This field is used in conjunction with the PVR Server **Retry Mount Time Limit**. If the number of consecutive mount errors which occur to any drive in this PVR equal or exceed this value, the drive is automatically locked by the PVL. The only mount errors that apply are those set through the **Retry Mount Time Limit** mechanism. The **Drive Error Count** field in the PVL Drive Information records the number of consecutive errors on a drive by drive basis. To turn off the automatic drive disable feature, set the **Drive Error Limit** to 0 or -1. Changing its value will not change drive disable behavior until the PVL is recycled.

ACSLS Packet Version. The packet version used by STK's ACSLS software.

Alternate SSI Port. A site running two SSI (Server System Interface) services on the same platform will specify the non-default port number of the second service via this field. Refer to Section 5.1.1.1.1: *STK PVR Additional Information* on page 121.

Dismount Delay. When **Defer Dismounts** is checked, this value is used by the PVL to determine the number of minutes that dismounts are delayed after the last data access.

Shelf Tape Check-In Retry. The number of seconds the PVR will wait before asking the robot if a requested shelf tape has been checked-in. The PVR will continue checking at this interval until the tape is checked-in. This field applies only if the **Support Shelf Tape** checkbox is selected. The retry value must be 30 or greater.

Shelf Tape Check-In Alarm. The PVR will periodically log alarm messages when a requested shelf tape has not been checked-in. This field specifies the number of minutes between alarms. This field is only active if the **Support Shelf Tape** checkbox is selected. The alarm value must be 2 or greater.

Characteristics. Flags for the PVR:

- **Defer Dismounts.** If ON, the PVL will delay the dismounting of a tape cartridge until the drive is required by another job or until the **Dismount Delay** time limit is exceeded.
- **Support Shelf Tape.** If ON, the PVR and the PVL will support the removal of cartridges from the tape library using the shelf_tape utility.

5.1. Real-Time Monitoring (RTM)

Real-time Monitoring (RTM) is a facility built into the Gatekeeper, the Core Server, and the Mover. Whenever an HPSS thread acting on behalf of a user job or a high-level system activity needs to block or process an operation that may take some time to complete, it will create an RTM record with information about what it is currently doing or waiting for. The RTM utility (rtmu) can be used to query the list of the current requests and obtain pertinent data for each request. The RTM requests can be used to construct an overview of the current operations being processed by the HPSS system as well as to monitor the overall health and status of the system. RTM requests that remain in the system for an extended period of time may indicate that the system is under heavy load. If a request remains for an unusually long time, it may indicate a hang condition or a deadlock situation that may require special intervention.

HPSS administrators and operators may use SSM to view the active RTM requests. The *RTM Summary* window lists a summary of the current RTM requests. The *RTM Detail* window displays detailed information for selected RTM requests.

5.1.1. RTM Summary List

Field Descriptions

RTM Summary List.

This is the main portion of the window which displays various information about each RTM request summary.

ID. The RTM request identifier.

Action. The action or operation that this request is currently executing. Examples include "Mover write", "PVL verify", "write tm"[1], etc.

State. The state of the requested operation. Examples include "in progress", "suspended", "blocked", etc.

Time Created. Of all the RTM request records with this **ID**, this field holds the age of the oldest. The time delta is the age, or time difference between when the request entered the system and the *RTM Summary List* window was last updated.

Time Updated. Of all the RTM requests records with this **ID**, this field holds the age of the request most recently updated by its server. The time delta is the age, or time difference between when the request was updated by the server and the *RTM Summary List* window was last updated.

Create Server. The server holding the RTM request with the oldest age or **Time Created**.

Update Server. The server holding the RTM request with the most recent update age or **Time Updated.**

Rec Cnt. The number of RTM records with this **ID.**

User Name. The printable username associated with this request (if available).

Path Name. The HPSS pathname of the file associated with this request (if available).

Wait Reason. A text string indicating what the request is currently waiting for. It may also contain additional information on the resources being waited for.

PVName. The name of the Physical Volume associated with this request (if available).

1 "write tm" means to write a tape mark.

5.1.2. RTM Detail

The *RTM Detail* window displays a snapshot of the details of the selected RTM requests from the *RTM Summary List* window. This may contain information from multiple servers, Gatekeeper, Core and Mover. The actual data displayed will be different for each server type and is displayed in a tree structure. Each node of the tree can be expanded/collapsed by clicking the mouse on the tree node indicator. A new snapshot can be taken and added to the display by pressing the **Snapshot** button. The tree can be cleared by pressing the **Clear** button.

Each detailed snapshot taken for an RTM request will contain the date and time the snapshot was taken at the top of the tree. Below the date and time there will be an **rtm_req_detail_entry_t.**

The fields contained in the **rtm_req_detail_entry_t** structure are;

> **ReqId.** The RTM request identifier.
>
> **UserName.** The printable username associated with this request (if available).
>
> **PathName.** The HPSS pathname of the file associated with this request (if available).
>
> **Oldest_StartTimeDelta.** Of all the RTM request records with this **ReqId**, this field holds the age of the oldest when the snapshot was taken.
>
> **Newest_UpdateTimeDelta.** Of all the RTM requests records with this **ReqId**, this field holds the age of the request most recently updated by its server when the snapshot was taken.
>
> **Oldest_ServerName.** The server holding the RTM request with the oldest age or **Oldest_StartTimeDelta**.
>
> **Newest_ServerName.** The server holding the RTM request with the most recent update age or **Newest_UpdateTimeDelta**.

Each server type that generates RTM request records (Gatekeeper, Core and Mover) has a server specific RTM data structure which will be displayed as a tree node below the **rtm_req_detail_entry_t** node. These structures are only displayed if there is a server of that type involved in the RTM request. There will be multiple entries if there are more than one server involved in the RTM request.

The server specific RTM records are called;

> rtm_gk_detail_entry_t (Gatekeeper)
>
> rtm_core_detail_entry_t (Core)
>
> rtm_mvr_detail_entry_t (Mover)

The server specific data structures each have a **Common** and **ReqInfo** structure embedded in them. The **Common** structure will contain the same fields for each server type.

The **Common** fields are;

ReqId. The RTM request identifier.

ReqCode. The action or operation that this request is currently executing. Examples include "Mover write", "PVL verify", "write tm" (tape mark), etc.

ReqState. The state of the requested operation. Examples include "in progress", "suspended", "blocked", etc.

ServerDescName. The descriptive name of the server that holds this RTM request record.

StartTimeDelta. The age of this request since it entered the server. That is, the time difference between when the request entered and when this snapshot was taken.

UpdateTimeDelta. The time difference between when this server last updated this RTM record and when this snapshot was taken.

ThreadId. The ID of the thread processing this request, useful in attaching to the thread to examine its state in the case of a hung request.

ProcessId. The ID of the UNIX process that is processing this request, useful in attaching to the process to examine its state in the case of a hung request.

WaitString. A text string indicating what the request is currently waiting for inside this server.

The **ReqInfo** portions of the server specific data structures will contain different fields depending on the server type. Not all fields for a particular server specific **ReqInfo** structure will be displayed. Only those fields that have valid data in them for the particular state of the RTM request will be displayed. The possible fields for each server specific **ReqInfo** are listed below.

The Core server **ReqInfo** structure may contain any or none of the following fields;

SubsystemId. The ID of the HPSS subsystem owning this request.

UserRealmId. The Realm ID of the user associated with this request.

FilesetId. The ID of the fileset of the file associated with this request.

FilesetName. The text string name of the fileset of the file associated with this request.

BfId. The Bitfile ID of the file associated with this request.

BfPathname. The HPSS pathname of the file associated with this request.

UserId. The ID of the user associated with this request.

UserGId. The Group ID of the user associated with this request.

COSId. The Class Of Service ID of the file associated with this request.

BFS_SClassId. The Storage Class ID known to the bitfile server of the file segment associated with this request.

BFS_BytesRequested. The number of bytes to be moved in this request as known by the bitfile server.

BFS_BytesMoved. The number of bytes already moved for this request as known by the bitfile server.

PVLJobId. The ID of the PVL job associated with this request.

MvrId. The ID of the Mover this request is currently waiting on.

DeviceId. The Device this request is curretly waiting on to complete a data move operation.

Segment. The ID of the storage segment being operated on by this request.

VV. The ID of the virtual volume associated with this request.

PVName. The name of the Physical Volume associated with this request.

CurrentRelPosition. The current media position of this physical volume as a relative address.

CurrentAbsPosition. The current media position of this physical volume as an absolute address.

ExcessVVsInUse. The Tape virtual volumes in use by this storage class have exceeded its quota. Waiting for a this number of virtual volumes to become available.

FamilyId. The ID of the File Family of the file associated with this request.

SS_SClassId. The Storage Class Id known to the Storage Server of the file/segment associated with this request.

SS_BytesRequested. The number of bytes to be moved by this request as known by the storage server.

SS_BytesMoved. The number of bytes that have been moved for this request as known by the storage server.

The Mover **ReqInfo** structure may contain any or none of the following fields;

InitialTransferOffset. The offset of the start of the mover device for the transfer associated with this request.

CurrentTransferOffset. The current offset of the mover device for the transfer associated with this request.

DeviceId. The ID of the device associated with this request.

VolumeId. The name of the volume associated with this request.

BytesRequested. The number of bytes the mover has been asked to transfer for this request.

BytesMoved. The number of bytes the mover has already transferred for this request.

The Gatekeeper **ReqInfo** structure may contain any or none of the following fields;

AuthorizedCaller. Indicates whether the caller is authorized or not.

BitFileId. The Bitfile ID of the file associated with this request.

ConnectionId. The ID of the (core server) connection associated with this request.

ClientConnectedId. The ID of the end client connection associated with this request.

ControlNo. A unique control ID used by the GateKeeper server.

RealmId. The Realm ID of the user associated with this request.

GroupId. The Group ID of the user associated with this request.

HostAddr. The address of the originating host

RequestType. The type of this request (Open, Create or Stage).

Oflag. The Open flags associated with this file open request.

StageFlags. Flags associated with this file stage operation.

StageLength. The number of bytes to stage.

StageOffset. The offset of the file where the stage is to begin.

StageStorageLevel. The Storage Class level that the file will be staged to.

UserId. The ID of the user associated with this request.

WaitTime. The length of time this request has been waiting.

5.2. Starting HPSS

This section describes how to start and stop HPSS servers and prerequisite software.

5.2.1. Starting HPSS Prerequisite Software

Invoke the /opt/hpss/bin/rc.hpss script as root to start the HPSS prerequisite software such as DB2, Kerberos and LDAP. By default, stdout and stderr from commands started by rc.hpss are not manipulated. Specifying the "-o" option to rc.hpss will cause stdout and stderr of commands invoked within the script to be sent to /dev/console, the terminal window where rc.hpss is running. When using the "-o" option, one should ensure that the HPSSLOG environment variable is not set to "stdout" in the env.conf file since this is likely to severely degrade HPSS' performance.

To start the prerequisite software, use the “-p” option to rc.hpss:

```
% su -
% /opt/hpss/bin/rc.hpss -p [start]
```

5.2.2. Starting HPSS Servers

Most HPSS servers are started via SSM which calls an appropriate Startup Daemon to begin execution of the selected HPSS server. This means that SSM and the Startup Daemons must be started in a different way. This section covers how to start the Startup Daemons, how to start SSM, and how to start the other HPSS servers once SSM and the Startup Daemons are up and running.

5.2.2.1. Starting the Startup Daemons

Invoke the /opt/hpss/bin/rc.hpss script as root to start the HPSS Startup Daemon. By default, both stdout and stderr are not redirected to /dev/console since this can have a negative impact on performance. Specifying the “-o” option will cause this redirection of output.

If server output is to be redirected to /dev/console, ensure that the HPSSLOG variable in the env.conf file is not set to stdout since this combination may severely degrade the HPSS performance.

To start the Startup Daemon, use the "-d" option to rc.hpss:

```
% su -
% /opt/hpss/bin/rc.hpss -d [start]
```

5.2.2.2. Starting SSM

The SSM System Manager configuration metadata should have already been created by **mkhpss** as part of the the infrastructure configuration. After SSM is started, this configuration metadata may be modified if necessary. Refer to the *HPSS Installation Guide,* Section 2.3.4: *HPSS Infrastructure* for more information.

The SSM server startup script, **rc.hpss**, can be used to invoke the HPSS System Manager. The server requires a number of environment variable values to be used as its command line arguments and to be used to configure and manage the HPSS servers. These variables are defined by the system. The system defined values can be overridden in the **env.conf** file. They should have been edited to reflect the site's configuration during the HPSS infrastructure configuration phase.

Before starting up the System Manager, ensure that DB2 and other prerequisite software are up and running.

The rc.hpss script will not start the System Manager if it is already running on the same node. Ensure that only one instance of the SSM System Manager is running at any one time.

To start the SSM System Manager, use the "-m" option to rc.hpss:

```
% su -
% /opt/hpss/bin/rc.hpss -m [start]
```

To start an SSM graphical user interface or command line interface session, see Section 3.3: *Configuration and Startup of hpssgui and hpssadm* on page 34, or the hpssgui or hpssadm man page. For best performance, it is recommended that the SSM graphical user interface (GUI) be installed on the SSM user's desktop machine. However, authorized SSM users can invoke the hpssgui script directly from the HPSS server machine if desired. However, most users of the SSM command line interface (hpssadm) will most likely invoke the script on the server machine.

5.2.2.3. Starting Other HPSS Servers

Other servers are started using SSM or can be automatically started upon startup of the System Manager by using the Core Server **Auto Startup** feature. This feature can be set using the *Core Server Configuration* window. Refer to Section 5.1.1.1: *Execution Controls* on page 90.

When starting a server from SSM using the *Servers* window, one or more HPSS servers can be started by the same request. SSM forwards the server's configuration data to the appropriate HPSS Startup Daemon. The Startup Daemon invokes the server with the appropriate input arguments. After the server is started, SSM attempts to establish a connection with the server. The server status is reported in the **Status** field on the *Servers* window. Refer to Section 5.1: *Server List* on page 83 for more information.

Before starting a server, ensure that DB2 and other prerequisite software are running, Kerberos is running (if using Kerberos authentication), that the Startup Daemon on the server's node is running and has a connection with SSM, and that the server is configured and its **Executable** flag is set. Ensure that any required PVR controllers are up and running before bringing up the PVRs.

To start a server, select the desired server(s) from the *Servers* window and click on the **Start** button. Verify the result of the request in the message area on the *Servers* window. In addition, monitor the *Alarms and Events* window for the "Server Initialized" event. Reference Section 5.2.2.3: on page 151.

The Startup Daemon allows only one instance of a configured server to be brought up at a time. If a server is already running, the subsequent startup request for that server will fail.

To be able to receive alarms and events associated with the server start up, it is recommended that the Log Daemon and all Log Clients be brought up first.

5.2.2.4. Automatic Server Restart

HPSS servers may be configured to automatically restart. An **Auto Restart Count** is contained in each server configuration. This count may be used to define the number of times a failed server will be automatically restarted without any administrative action. The **Auto Restart Count** is ignored for the Startup Daemon and the SSM System Manager.

A failure is defined as a server terminating abnormally, and an abnormal termination is the result of the server exiting with a nonzero return code or as a result of a signal other than SIGTERM or SIGKILL. Termination due to an administrative halt or shutdown is not considered to be an abnormal termination and a server will not be automatically restarted.

The following **Auto Restart Count** values may be set:

Table 1. Auto Restart Count Values

Auto Restart Count Value	**Meaning**
0	Do not restart
-1	Infinite restart attempts
n	Restart a failed server n times (where n is > 0)

After the **Auto Restart Count** is exhausted, the Startup Daemon will not attempt to restart the server. If the server executes for more than 1 hour without failing, the number of failures will be set back to 0.

> *Do not confuse this feature with the Core Server* **Auto Startup** *feature. This feature can be set using the Core Server Configuration window described in* Section 5.1.1.1: *Execution Controls* on page 90.

5.1. Stopping HPSS

5.1.1. Shutting Down an HPSS Server

One or more HPSS servers can be shut down in a single SSM request. SSM will send each server a request to perform a controlled shut down. The server will attempt to complete its current tasks, discontinue accepting new tasks, and inform SSM of its termination prior to exiting.

To initiate the request, select the desired server(s) on the *Servers* window and click on the **Shut Down** button. SSM will confirm the request through a pop-up window. The SSM user may wish to verify the result of the request in the status bar on the *Servers* window as well as to monitor the *Alarms and Events* window for the “Server Terminated” event.

> *The HPSS Startup Daemon(s) and the SSM System Manager cannot be shut down from the Servers window. Select **System Manager** from the **Shutdown** submenu of the **Operations** menu of the Health and Status window to shut down the System Manager. Use the **rc.hpss** script stop option to shut down either the System Manager or the Startup Daemon.*

Servers may not terminate immediately since they may wait for pending operations to complete before terminating. Monitor the *Alarms and Events* window for possible notifications of delay from the server. In addition, monitor the *Servers* window for the server’s shut down status. If an immediate shut down is required, use the **Force Halt** operation described in Section 5.1.3: *Halting an HPSS Server* on page 152.

SSM must have a connection to the startup daemon associated with the server or the shutdown operation will fail.

5.1.2. Shutting Down All HPSS Servers

Most HPSS servers (all but SSM and the Startup Daemon) can be shut down by selecting **Shut Down All Non-SSM Servers** from the **Shutdown** submenu of the **Operations** menu. SSM will process this request similarly to the **Shutdown Server** request issued from the *Servers* window.

This feature is useful when the entire HPSS system must be shut down, leaving SSM up to monitor the server shut down process. When the servers are down, the SSM server and sessions can be shut down by selecting **System Manager** from the **Shutdown** submenu of the **Operations** menu. The SSM client sessions will detect that the SSM System Manager has been shutdown and will then inform the user of the event.

5.1.3. Halting an HPSS Server

Halting an HPSS server stops it without giving it an opportunity to shut down gracefully. When the command is received by the server, the server simply terminates immediately without doing any of its normal shutdown processing. It should be used only if it is necessary to shut down the server as quickly as possible or if attempts to shut the server down using the **Shutdown** button have failed.

One or more HPSS servers can be halted in a single request. SSM will issue a request to each selected server's Startup Daemon to issue a SIGKILL signal to its HPSS server.

To initiate the request, select the desired server(s) on the *Servers* window and click on the **Force Halt** button. SSM will confirm the request through a pop-up window. Verify the result of the request in the status bar on the *Servers* window. In addition, monitor the *Alarms and Events* window for the “Server Halted” alarm.

The HPSS Startup Daemon(s) and the SSM System Manager cannot be halted from the *Servers* window.

SSM must have a connection to the Startup Daemon associated with the server or the Halt Server operation may fail.

5.1.4. Shutting Down the SSM Server

To shutdown the SSM server, select **System Manager** from the **Shutdown** submenu of the **Operation** menu on the *HPSS Health and Status* window. All SSM user sessions, including graphical and command line sessions, will detect that the System Manager has exited. Choosing this option will pop up a confirmation window which allows the shutdown request to be approved or canceled.

As the System Manager exits, a notification window will pop up on each logged on SSM graphical user interface session informing the user that the GUI has lost connection to the System Manager. The session user can click the **Exit** button on this window to exit the SSM session or the **Login** button to log out of the session and get a new login window, which can be used once the System Manager has been restarted.

At each command line user interface session, a notification message will be displayed that the client connection to the System Manager has been lost and the session will be terminated.

These notifications will also happen whenever the client is having communication problems with the System Manager, even if the System Manager is still running.

Additionally the System Manager can be shut down with the "-m" option of rc.hpss:

```
% su -

% /opt/hpss/bin/rc.hpss -m stop
```

5.1.5. Shutting Down the Startup Daemon

To stop the Startup Daemon, use the "-d" option of rc.hpss:

```
% su -
% /opt/hpss/bin/rc.hpss -d stop
```

5.1.6. Stopping the Prerequisite Software

To stop the prerequisite software:, use the "-p" option of rc.hpss:

```
% su -
% /opt/hpss/bin/rc.hpss -p stop
```

This will stop Kerberos, LDAP and DB2 if the corresponding shutdown scripts exist in /var/hpss/etc.

5.2. Server Repair and Reinitialization

This section describes how to repair or reinitialize an HPSS server.

5.2.1. Repairing an HPSS Server

Repairing a server is the process of instructing the server to reset its internal state and status variables to their nominal settings. These variables include Operational State, Usage State, Administrative State, Service Status, Security Status, Software Status, Hardware Status and Communication Status. Operational State is displayed prominently on the *Servers* window. If a server encounters an error as it runs, it may set one or more of these state and status variables to a value indicating the type and severity of the error. Alarm messages may be generated as well. Repairing a server simply resets all of these state and status variables to their nominal settings. One or more servers can be repaired with each request.

To initiate the request, select the desired servers from the *Servers* window and click on the **Repair** button. Verify the result of the request in the status bar on the *Servers* window. In addition, monitor the *Alarms and Events* window for the “Server Repaired” event.

Repairing a server does not correct the underlying problem that caused the server's reported state to change. Rather, it is a means for the administrator to notify the server that the underlying problem has been corrected or dismissed. It is an “alarm reset”.

5.2.2. Reinitializing a Server

Reinitializing a server causes it to re-read all or part of its configuration metadata and reset its internal representation of those values. In many cases reinitialization is all that is needed to cause a server to start using a new configuration. In other cases it may be necessary to restart a server to cause it to use a new configuration.

HPSS servers vary widely in their support of the reinitialization function. The following table lists the behavior of each HPSS server when it is told to reinitialize.

Table 1. Server Reinitialization Behavior

Server	**Behavior**
Log Daemon, Log Client, Location Server	These servers support full reinitialization. This means that reinitialization is as effective as restarting the server.

Core Server	Resets the **COS Copy to Disk**, **COS Change Retry Limit**, **Tape Dismount Delay**, **Tape Handoff Delay**, **PVL Max Connection Wait**, **Fragment Trim Limit** and **Fragment Smallest Block** values to the values in the specific configuration metadata record. Reloads cached **Class of Service** information for those **COS**s that were already cached in the server's memory. Does not add new **COS**s to, or remove deleted **COS**s from the cache. Does not update **COS** cache information if the **Hierarchy ID** in the **COS** changes. Reloads cached **Migration Policy** for those policies that were already cached in the server's memory. Does not add new policies to, or remove deleted policies from the cache. Reloads cached global and subsystem **Metadata Space Warning Threshold**, **Metadata Space Critical Threshold**, **Metadata Space Monitor Interval** and the **DB Log Monitor Interval** values using values read from the Global and Subsystem configuration metadata. Reinitializes the Restricted User list in memory. Releases the in-memory cache of File Family Ids, allowing it to be reconstructed. If a file family has been removed from the list of defined families, this step causes it to be unknown to the Core Server. Reloads the in-memory cache of HPSS environment variables from the "env.conf" file. Any Core Server configuration information not listed above is not changed when the server is reinitialized.
Mover	Re-reads log policy only.
PVR	Re-initializes the internal drive list cache.
All other servers	No support for reinitialization.

To reinitialize a server, select the desired server(s) from the *Servers* window and click on the **Reinitialize** button. SSM will confirm the request through a pop-up window. The result(s) of the request(s) may be verified via the status bar on the *Servers* window as well through the *Alarms and Events* window.

If a server does not support reinitialization, a pop-up window will be displayed to inform the user of the invalid request. Warnings are not displayed for servers that only partially support reinitialization.

Servers that do not support reinitialization, or those that do not support reinitializing the settings in question, must be restarted in order for configuration modifications to take affect.

Some groups of servers depend on consistent configuration information to run properly. For example, the Core Server and Migration/Purge Server must agree on Class of Service, Hierarchy and Storage Class configurations. When changes are made to these sorts of configuration information, all of the servers that rely on these settings should be restarted to make sure they agree.

5.1. Forcing an SSM Connection

When the status of the server's connection with SSM is UP/UNCONNECTED, select the server and then click on the **Force Connect** button on the *Servers* window. SSM will attempt to establish the connection with the server and reflect the latest data in the **Status** field on the *Servers* window.

The SSM System Manager periodically retries the connection with all the servers that are configured to execute. The **Force Connect** button may be used to trigger an immediate connection request.

Chapter 6. Storage Configuration

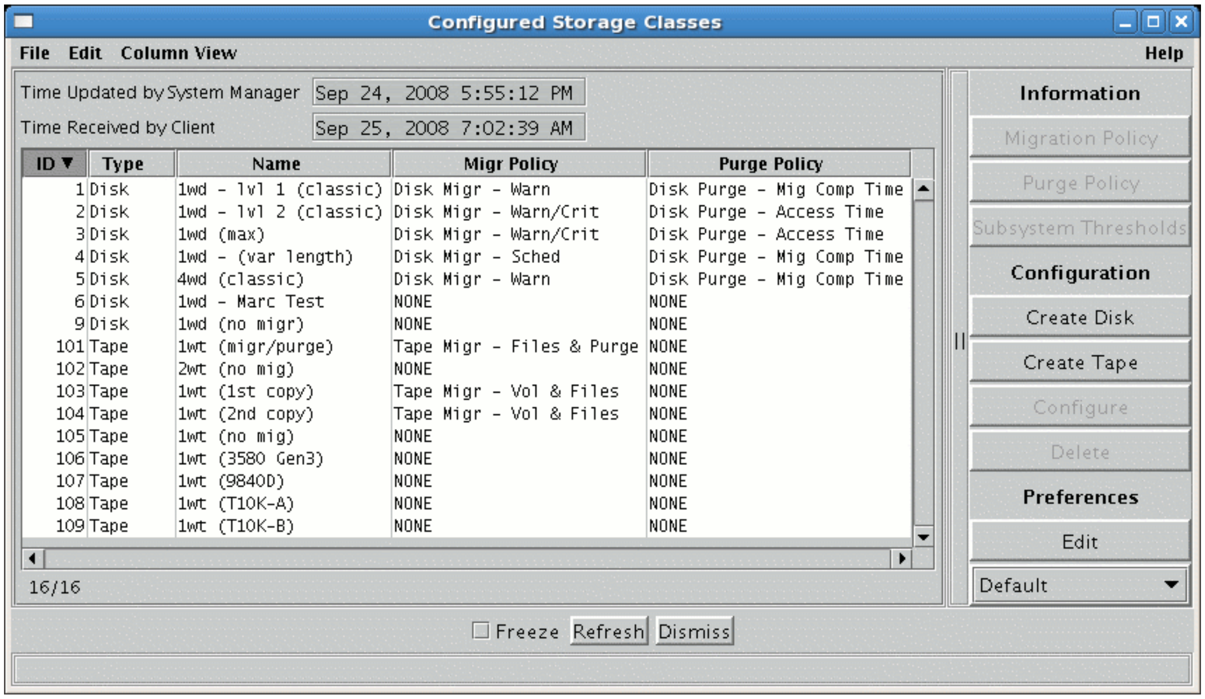
This chapter describes the procedures for creating, modifying, and deleting storage classes, hierarchies, classes of service, migration policies, purge policies, and file families.

6.1. Storage Classes

This section describes the process of configuring Storage Classes.

6.1.1. Configured Storage Classes Window

A storage class can be created and managed using the *Configured Storage Classes* window.



Time Updated by System Manager: Sep 24, 2008 5:55:12 PM

Time Received by Client: Sep 25, 2008 7:02:39 AM

ID ▼	Type	Name	Migr Policy	Purge Policy
1	Disk	1wd - lvl 1 (classic)	Disk Migr - Warn	Disk Purge - Mig Comp Time
2	Disk	1wd - lvl 2 (classic)	Disk Migr - Warn/Crit	Disk Purge - Access Time
3	Disk	1wd (max)	Disk Migr - Warn/Crit	Disk Purge - Access Time
4	Disk	1wd - (var length)	Disk Migr - Sched	Disk Purge - Mig Comp Time
5	Disk	4wd (classic)	Disk Migr - Warn	Disk Purge - Mig Comp Time
6	Disk	1wd - Marc Test	NONE	NONE
9	Disk	1wd (no migr)	NONE	NONE
101	Tape	1wt (migr/purge)	Tape Migr - Files & Purge	NONE
102	Tape	2wt (no mig)	NONE	NONE
103	Tape	1wt (1st copy)	Tape Migr - Vol & Files	NONE
104	Tape	1wt (2nd copy)	Tape Migr - Vol & Files	NONE
105	Tape	1wt (no mig)	NONE	NONE
106	Tape	1wt (3580 Gen3)	NONE	NONE
107	Tape	1wt (9840D)	NONE	NONE
108	Tape	1wt (T10K-A)	NONE	NONE
109	Tape	1wt (T10K-B)	NONE	NONE

16/16

This window displays all storage classes in the HPSS system, both empty and non-empty. This window is displayed by first selecting Configure from the *Health and Status* window. This will display a drop-down menu. Select Storage Space from this menu. This will display another drop-down menu. Select Storage Classes from this menu.

See also the related window *Active Storage Classes*, as described in Section 8.3.1: *Active Storage Classes Window* on page 248. The *Active Storage Classes* window does not list any empty storage classes, but only those to which volumes have been assigned.

Field Descriptions

Storage Class List. The central panel in this window shows the list of configured storage classes. The columns display selected fields from the *Storage Class Configuration* windows discussed in the following sections.

Information Buttons.

Migration Policy. Opens the configuration window for the migration policy that is configured for the selected storage class. This button will be disabled if no storage classes are selected in the Storage Classes list or the selected storage class does not have a migration policy.

Purge Policy. Opens the configuration window for the purge policy that is configured for the selected storage class. This button will be disabled if no storage classes are selected in the Storage Classes list or the selected storage class does not have a purge policy.

Subsystem Thresholds. Opens either a disk or tape subsystem threshold configuration window, as appropriate, for the selected storage class.

Configuration Buttons.

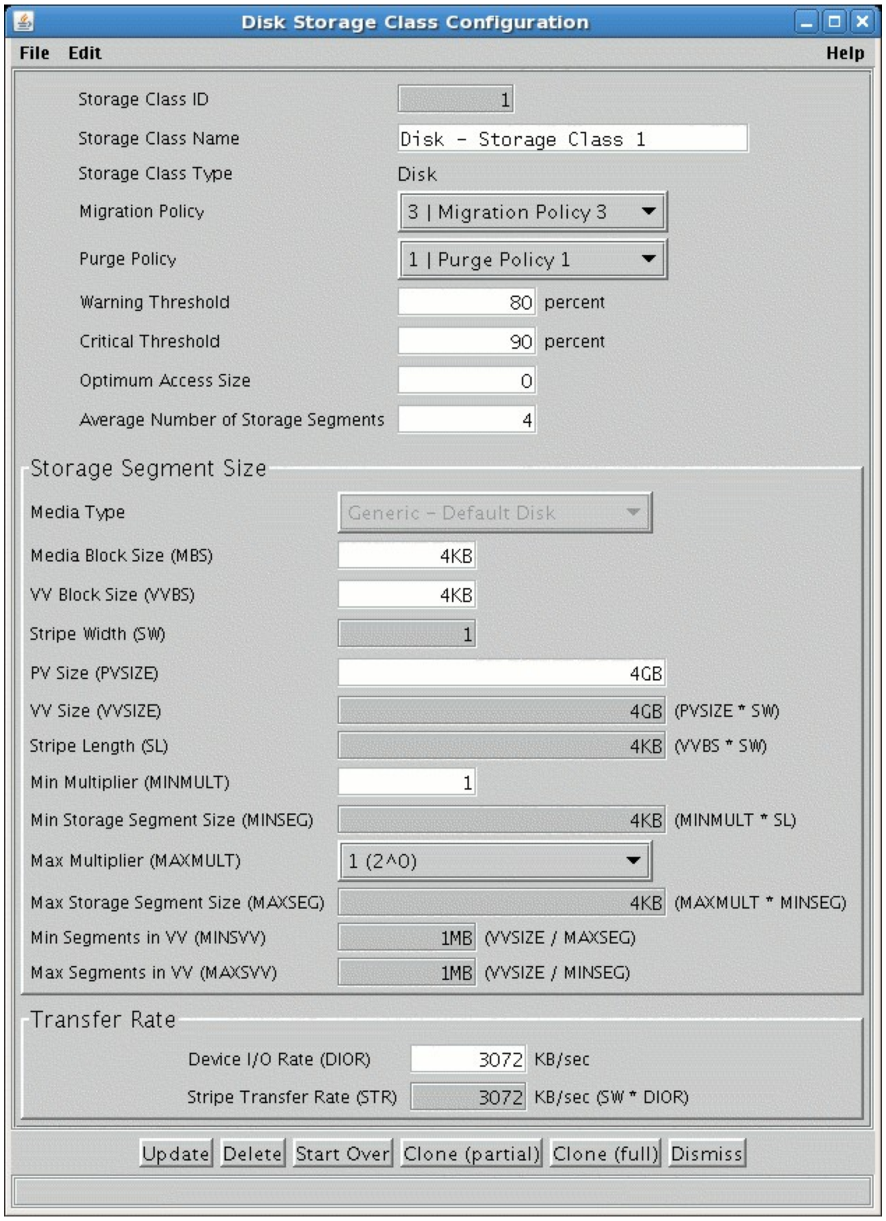
Create Disk. Opens a *Disk Storage Class* window containing default values for a new disk storage class.

Create Tape. Opens a *Tape Storage Class* window containing default values for a new tape storage class.

Configure. Opens the selected storage class configuration(s) for editing.

Delete. Deletes the selected storage class(es).

6.1.2. Disk Storage Class Configuration



Storage Class ID	1
Storage Class Name	Disk - Storage Class 1
Storage Class Type	Disk
Migration Policy	3 | Migration Policy 3
Purge Policy	1 | Purge Policy 1
Warning Threshold	80 percent
Critical Threshold	90 percent
Optimum Access Size	0
Average Number of Storage Segments	4

Storage Segment Size

Media Type	Generic - Default Disk	
Media Block Size (MBS)	4KB	
VV Block Size (VVBS)	4KB	
Stripe Width (SW)	1	
PV Size (PVSIZE)	4GB	
VV Size (VVSIZE)	4GB	(PVSIZE * SW)
Stripe Length (SL)	4KB	(VVBS * SW)
Min Multiplier (MINMULT)	1	
Min Storage Segment Size (MINSEG)	4KB	(MINMULT * SL)
Max Multiplier (MAXMULT)	1 (2^0)	
Max Storage Segment Size (MAXSEG)	4KB	(MAXMULT * MINSEG)
Min Segments in VV (MINSVV)	1MB	(VVSIZE / MAXSEG)
Max Segments in VV (MAXSVV)	1MB	(VVSIZE / MINSEG)

Transfer Rate

Device I/O Rate (DIOR)	3072	KB/sec
Stripe Transfer Rate (STR)	3072	KB/sec (SW * DIOR)

Update Delete Start Over Clone (partial) Clone (full) Dismiss

This window is used to manage disk storage class configurations.

Field Descriptions

Storage Class ID. The numeric identifier assigned to the storage class.

Storage Class Name. The descriptive name of the storage class.

Storage Class Type. The type of the storage class (Disk).

Migration Policy. The migration policy associated with this storage class, or None if no migration is desired.

Advice - Do not configure a migration policy for a storage class at the lowest level in a hierarchy.

*If a migration policy is added to a storage class after files are created in the storage class, those files may never be migrated. Use the **mkmprec** utility to correct this problem. See the **mkmprec** man page for more information.*

Purge Policy. The purge policy associated with this storage class, or None if no purge is desired.

Advice - Do not configure a purge policy for a tape storage class or for any storage class which does not have a migration policy in effect. Purge policies only apply to disk storage classes. Purging from tape is managed as a special characteristic of the tape migration policy.

Warning Threshold. A threshold for space used in this storage class expressed as a percentage of the total space defined in the storage class. Alarms will be sent to SSM periodically when free space in the storage class falls below this value, and the **Space Thresholds** field on the *HPSS Health and Status* window will be changed to Warning. Note that this field will not have any effect if overridden by Storage Subsystem-Specific Thresholds.

Critical Threshold. Another threshold for space used in this storage class expressed as a percentage of the total space defined in the storage class. Alarms will be sent to SSM periodically when free space in the storage class falls below this value, and the **Space Thresholds** field on the *HPSS Health and Status* window will be changed to Critical. Note that this field will not have any effect if overridden by Storage Subsystem-Specific Thresholds.

Advice - These values determine limits beyond which the MPS sends alarms indicating that storage space is running low. The thresholds on disk are measured in bytes and expressed in terms of the percentage of total space in the storage class. The threshold may be disabled by setting the value to 100 percent.

Optimum Access Size. The optimal transmission size to be used for a transfer request using this storage class. (Not currently used by HPSS. May be used by site specific applications)

Average Number of Storage Segments. The **Average Number of Storage Segments** is used by the Core Server as an aid in picking the size of disk storage segments when writing a file to disk. This field is ignored when the storage class is at the top level of the hierarchy in a COS which uses Variable Length or Fixed Length Max Style segment allocation. See the discussion of the COS **Allocation Method** field and **Truncate Final Segment** flag in Section 6.3.2 *Class of Service Configuration Window* on page 175 for details on segment allocation methods.

When the storage class is at the top level of a COS which uses Fixed Length Classic Style segment allocation, or when the storage class is at any level below the top, regardless of the COS segment allocation method, the Core Server requires that all of the disk storage segments occupied by a given file be the same size, and must be within the range of sizes given by the Minimum Storage Segment Size and Maximum Storage Segment Size parameters.

If **Max Storage Segment Size (MAXSEG)** is greater than **Min Storage Segment Size (MINSEG)**, the Core Server attempts to select a storage segment size for a file such that the file can be stored in **Average Number of Storage Segments** segments. If the **Max Storage Segment Size (MAXSEG)** is equal to the **Min Storage Segment Size (MINSEG)**, the Core Server will use this value for the storage segment size, and the file will occupy as many of these segments as necessary.

The principal effect of changing the value of the **Average Number of Storage Segments** is on disk volume fragmentation. If the files to be stored in a storage class are large relative to the size of the disk

VVs, fragmentation of the volumes may make it difficult to find space for a new segment. Setting **Average Number of Storage Segments** to a larger value will increase the number of segments occupied by files, and decrease the segment size. Fragmentation of the volumes will be reduced, but the amount of metadata required to describe the files will be increased.

Conversely, if files are small relative to the size of the disk VVs, smaller values of **Average Number of Storage Segments** increase the size of the storage segments, and decrease the number of segments occupied by each file. This reduces the metadata storage requirements of the file.

The number of segments in small HPSS files can have a significant impact on transfer performance. To maximize the transfer performance, set this value to 1. Keep in mind that this will result in less effective disk space utilization. On average, you will use only 50% of your disk space with this selection.

Storage Segment Size

Media Type. The media type of all volumes in the storage class.

Media Block Size (MBS). The **Media Block Size** is the size in bytes of a physical data block on the media. For disk, the value must be a multiple of the physical block size used by the disk hardware. For example, if the disk hardware stores data in 512-byte sectors, 2048 would be a valid entry in this field, but 2000 would not.

*Advice - The **Media Block Size** should be set to a value appropriate for the volume type. See the HPSS Installation Guide, Section 3.10.1.1: Media Block Size Selection for some recommendations.*

VV Block Size (VVBS). The virtual volume block size is the number of bytes written to an element of a striped VV before the data stream moves to the next stripe element. It can be thought of as the stride length of striped data transfer operations. The length of the VV block has an effect on the striping efficiency. Short VV blocks cause more protocol overhead when writing striped devices. In non-striped applications, VV Block Size has little meaning so any convenient multiple of the Media Block Size will do.

Advice - When choosing a VV Block Size, the administrator should consider the characteristics of any data source or sink that will be copied to or from. Best performance of striped copies usually occurs when the VV Block Sizes of the source and sink are equal. This minimizes the data movement protocol overhead and helps to keep the data streams flowing smoothly.

VV Block Size must be an integer multiple of the Media Block Size.

See the HPSS Installation Guide, Section 3.10.1.2: Virtual Volume Block Size Selection (disk) and Section 3.10.1.3: Virtual Volume Block Size Selection (tape) for information.

Stripe Width (SW). The number of Physical Volumes in a Virtual Volume in this Storage Class.

PV Size (PVSIZE). The size of the disk volume in bytes.

VV Size (VVSIZE). The virtual volume size. The product of the **PV Size (PVSIZE)** and the **Stripe Width (SW)**.

Stripe Length (SL). The **Stripe Length** is the product of the **VV Block Size (VVBS)** and the **Stripe Width (SW)**.

Min Multiplier (MINMULT). The **Min Storage Segment Size (MINSEG)** field is set to the product of this value and **Stripe Length (SL)**.

Min Storage Segment Size (MINSEG). The lower bound for storage segment sizes created on volumes in this storage class. This value is the product of the **Stripe Length (SL)** and the **Min Multiplier (MINMULT)**.

Max Multiplier (MAXMULT). The **Max Storage Segment Size (MAXSEG)** must be a power of 2 multiple of the **Stripe Length (SL)**. This selection list contains the valid power of 2 values from 1 (2^0) to 16,777,216 (2^24). Select the appropriate multiplier from the selection list. The field **Max Storage Segment Size (MAXSEG)** is set to the product of this value and **Stripe Length (SL)**.

Max Storage Segment Size (MAXSEG). The upper bound for storage segment sizes created on volumes in this storage class. This must be a power of 2 multiple of the **Stripe Length (SL)**. This value is the product of the **Stripe Length (SL)** and the **Max Multiplier (MAXMULT)**.

Min Segments in VV (MINSVV). The minimum number of segments in a virtual volume. This value is calculated as the **VV SIZE (VVSIZE)** parameter divided by the **Max Storage Segment Size (MAXSEG)** parameter.

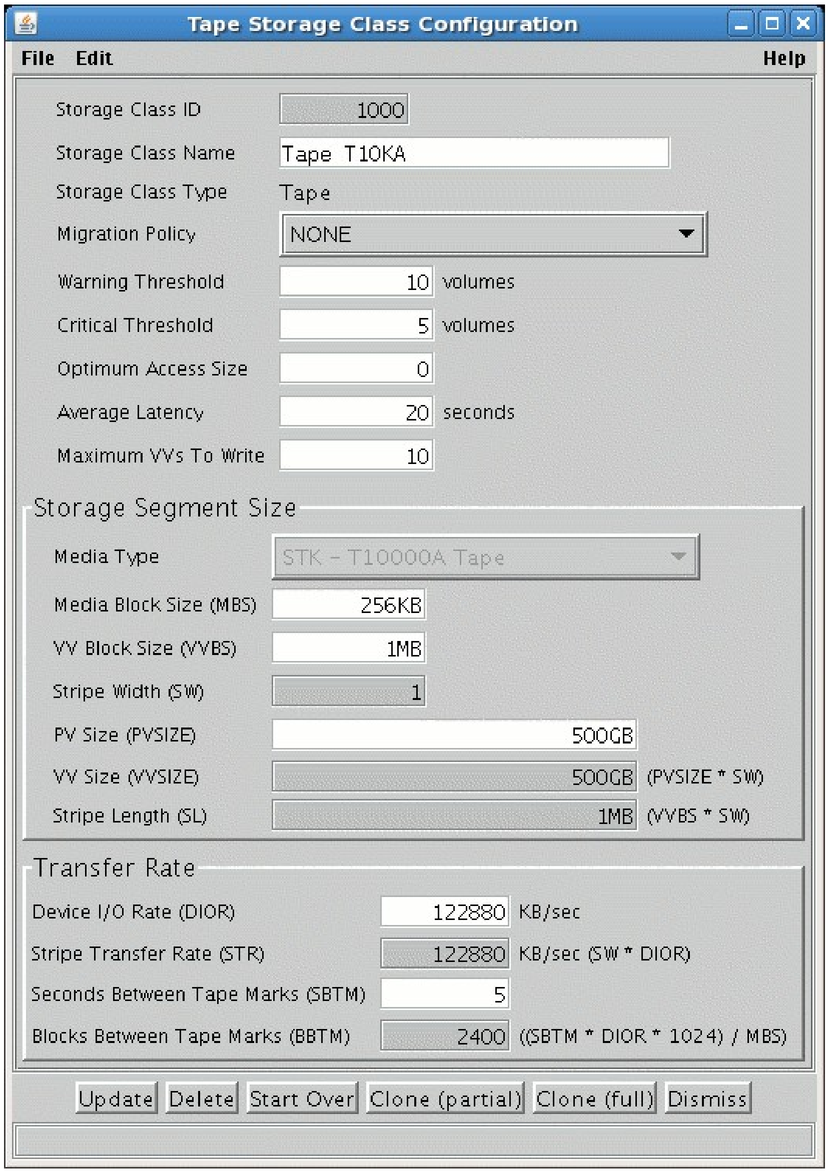
Max Segments in VV (MAXSVV). The maximum number of segments in a virtual volume. This value is calculated as the **VV SIZE (VVSIZE)** parameter divided by the **Min Storage Segment Size (MINSEG)** parameter.

Transfer Rate

Device I/O Rate (DIOR). The approximate data transfer speed, in kilobytes per second (KB/sec), which can be achieved by devices corresponding to **Media Type**. In disk storage classes, this field is informational only.

Stripe Transfer Rate (STR). The estimated aggregate transfer rate for volumes of this type. This value is the product of the **Stripe Width (SW)** and the **Device I/O Rate (DIOR)**.

6.1.3. Tape Storage Class Configuration


Tape Storage Class Configuration

Storage Class ID	1000	
Storage Class Name	Tape T10KA	
Storage Class Type	Tape	
Migration Policy	NONE	
Warning Threshold	10	volumes
Critical Threshold	5	volumes
Optimum Access Size	0	
Average Latency	20	seconds
Maximum VVs To Write	10	

Storage Segment Size

Media Type	STK - T10000A Tape	
Media Block Size (MBS)	256KB	
VV Block Size (VVBS)	1MB	
Stripe Width (SW)	1	
PV Size (PVSIZE)	500GB	
VV Size (VVSIZE)	500GB	(PVSIZE * SW)
Stripe Length (SL)	1MB	(VVBS * SW)

Transfer Rate

Device I/O Rate (DIOR)	122880	KB/sec
Stripe Transfer Rate (STR)	122880	KB/sec (SW * DIOR)
Seconds Between Tape Marks (SBTM)	5	
Blocks Between Tape Marks (BBTM)	2400	((SBTM * DIOR * 1024) / MBS)

This window is used to manage tape storage class configurations.

Field Descriptions

Storage Class ID. The numeric identifier assigned to the storage class.

Storage Class Name. The descriptive name of the storage class.

Storage Class Type. The type of the storage class (Tape).

Migration Policy. The migration policy associated with this storage class, or None if no migration is desired.

Advice - Do not configure a migration policy for a storage class at the lowest level in a hierarchy.

*If a migration policy is added to a storage class after files are created in the storage class, those files may never be migrated. Use the **mkmprec** utility to correct this problem. See the **mkmprec** man page for more information.*

Warning Threshold. A threshold for space used in this storage class expressed as a number of empty tape volumes. Alarms will be sent to SSM periodically when the number of empty tapes in the storage class falls below this value, and the **Space Thresholds** field on the *HPSS Health and Status* window will be changed to Warning. Note that this field will not have any effect if overridden by Storage Subsystem-Specific Thresholds.

Critical Threshold. Another threshold for space used in this storage class expressed as a number of empty tape volumes. Alarms will be sent to SSM periodically when the number of empty tapes in the storage class falls below this value, and the **Space Thresholds** field on the *HPSS Health and Status* window will be changed to Critical. Note that this field will not have any effect if overridden by Storage Subsystem-Specific Thresholds.

Optimum Access Size. The optimal transmission size to be used for a transfer request using this storage class. (Not currently used by HPSS. May be used by site specific applications)

Average Latency. The average time (in seconds) that elapses when a data transfer request is scheduled and the time the data transfer begins.

Maximum VVs to Write. The number of tape virtual volumes in the storage class that a Core Server will use for concurrent writes.

Advice - Small values in this field restrict files being written in the storage class to a small number of tapes, reducing the number of tape mounts. The number of tape drives used to write files in the storage class will be limited to approximately the value of this field times the stripe width of the mounted VVs. Read operations are not limited by this value.

Storage Segment Size

Media Type. The media type associated with the storage class.

Media Block Size (MBS). The Media Block Size is the size in bytes of a physical data block on the media. For tape, this can be almost any value within reasonable limits. If the tape hardware has a recommended physical block size, use that value.

*Advice - The **Media Block Size** should be set to a value appropriate for the volume type. See the HPSS Installation Guide, Section 3.10.1.1: Media Block Size Selection for some recommendations.*

See also Device I/O Rate and Seconds Between Tape Marks in this section.

VV Block Size (VVBS). The virtual volume block size is the number of bytes written to an element of a striped VV before the data stream moves to the next stripe element. It can be thought of as the stride length of striped data transfer operations. The length of the VV block has an effect on the striping efficiency. Short VV blocks cause more protocol overhead when writing striped devices. In non-striped applications, **VV Block Size** has little meaning so any convenient multiple of the **Media Block Size** will do.

Advice - When choosing a VV Block Size, the administrator should consider the characteristics of any data source or sink the storage class that will be copied to or from. Best performance of striped copies usually occurs when the VV Block Sizes of the source and sink are equal. This minimizes the data *movement protocol overhead and helps to keep the data streams flowing smoothly.*

VV Block Size must meet the following constraining requirements:

- It must be an integer multiple of the Media Block Size.
- The PV Section Length (**Media Block Size (MBS) * Blocks Between Tape Marks (BBTM**) divided by the **VV Block Size (VVBS)** must be a whole number. For example, if the **Media Block Size (MBS)** is 64 KB, and the **Blocks Between Tape Marks (BBTM)** is 512, the physical volume section length is 32 MB. The **VV Block Size (VVBS)** could be 64 K, 128 K, 256 K, or larger, but not 192 K.

See the HPSS Installation Guide , Section 3.10.1.2: Virtual Block Size Selection (disk) and Section 3.10.1.3:Virtual Block Size Selection(tape) for information.

Stripe Width (SW). The number of Physical Volumes in a Virtual Volume in this Storage Class.

PV Size (PVSIZE). The estimated size of the tape volume in bytes. The actual number of bytes that can be written to a tape will vary with individual media and data compressibility. Think of this field as the nominal length of a tape of this type.

VV SIZE (VVSIZE). The virtual volume size. The product of the **PV Size (PVSIZE)** and the **Stripe Width (SW)**.

Stripe Length (SL). The **Stripe Length (SL)** is the product of the **VV Block Size (VVBS)** and the **Stripe Width (SW)**.

Transfer Rate

Device I/O Rate (DIOR). The approximate data transfer speed, in kilobytes per second (KB/sec), which can be achieved by devices corresponding to Media Type.

*Advice - This field is used in calculating a reasonable setting for **Seconds Between Tape Marks (SBTM)**.*

Stripe Transfer Rate (STR). The estimated aggregate transfer rate for volumes of this type. This field is calculated using the **Stripe Width (SW)** times the **Device I/O Rate (DIOR)**.

Seconds Between Tape Marks (SBTM). The number of seconds between tape mark writes while writing continuously to tape.

*Advice - The **Device I/O Rate (DIOR)**, the **Media Block Size (MBS)** and the **Seconds Between Tape Marks (SBTM)** determine the length of a physical volume section. The administrator should set the **Device I/O Rate (DIOR)** to an appropriate value for the device, and then pick a **Media Block Size (MBS)**. Finally, adjust the **Seconds Between Tape Marks (SBTM)** to strike a balance between tape utilization and transfer efficiency.*

*Smaller values of **Seconds Between Tape Marks (SBTM)** cause more tape to be consumed in tape marks and break up data streaming throughput on streaming tape drives, but cause controller buffers to be flushed more frequently and may improve access times to the middle of large files.. Larger values improve transfer efficiency. However, smaller values may reduce the amount of data that may need to be re-written on another tape when a tape reaches EOM. As tape drive streaming rates increase, we generally recommend larger values for this setting.*

If the tape media supports "fast locate", and that feature is enabled for the tape devices, choose larger values of ***Seconds Between Tape Marks (SBTM)****. When reading from the middle of a file on tape, the fast locate feature is used by HPSS to locate the data block in which a given portion of a file is located, rather than skipping tape marks and data blocks. When fast locate is enabled, there is no advantage to using smaller values of* ***Seconds Between Tape Marks (SBTM)*** *for locating positions within files. The fast locate feature is also know as Logical Block Addressing (LBA).*

Blocks Between Tape Marks (BBTM). The maximum number of data blocks (of **Media Block Size (MBS)**) that will be written on a tape between consecutive tape marks.

The number of media data blocks between tape marks is calculated by SSM when the **Device I/O Rate (DIOR)** and **Seconds Between Tape Marks (SBTM)** are set. This value is displayed for information only.

The recording space between tape marks is called a "section". The PV Section Length is **Blocks Between Tape Marks (BBTM)** times the **Media Block Size (MBS)**. The VV Section Length is the PV Section Length times the **Stripe Width (SW)**.

6.1.4. Storage Class Subsystem Thresholds

These windows allow the user to manage storage class space thresholds which are specific to a storage subsystem. These windows are accessible only from the *Configured Storage Classes* window. The user must first select a storage class from the storage classes list. The selected storage class will be referred to as the 'parent' storage class to the subsystem threshold window which is opened when the storage class is selected.

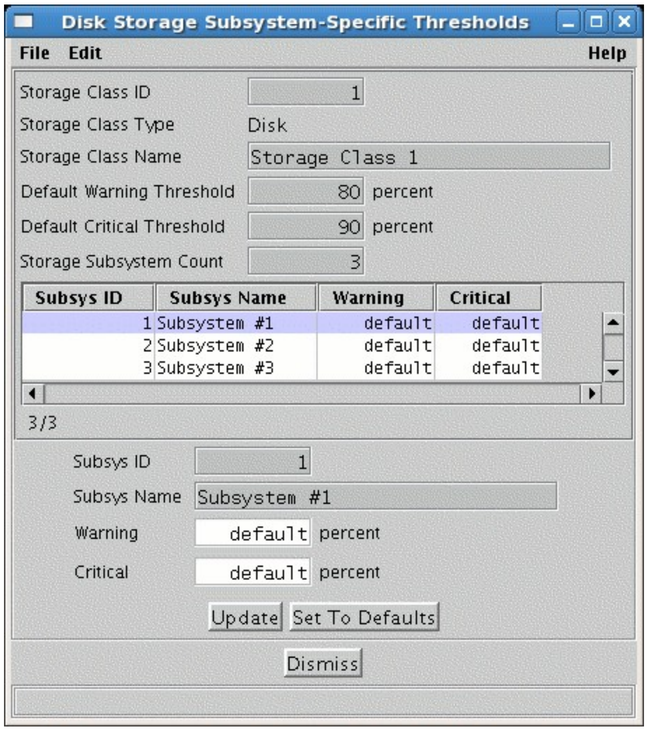
Each *Storage Class Configuration* window has fields for **Warning Threshold** and **Critical Threshold**. By default, these thresholds apply to all storage subsystems using the storage class. However, the storage class thresholds can be overridden by configuring a subsystem-specific threshold allowing a storage subsystem to have its own customized set of **Warning Threshold** and **Critical Threshold** values.

After creating or modifying any thresholds, the Core Server and Migration Purge Server for the affected subsystem must be recycled to make the changes effective.

Related Information

Section 4.2.3.8: *Storage Class Threshold Overrides* on page 81.

6.1.4.1. Disk Storage Subsystem-Specific Thresholds



Storage Class ID	1
Storage Class Type	Disk
Storage Class Name	Storage Class 1
Default Warning Threshold	80 percent
Default Critical Threshold	90 percent
Storage Subsystem Count	3

Subsys ID	Subsys Name	Warning	Critical
1	Subsystem #1	default	default
2	Subsystem #2	default	default
3	Subsystem #3	default	default

3/3

Subsys ID	1
Subsys Name	Subsystem #1
Warning	default percent
Critical	default percent

This window is used to define Warning and Critical thresholds unique to a particular storage subsystem, overriding the values defined in the disk storage class. The user may modify either the **Warning** percent, **Critical** percent or both for one or more of the listed subsystems. Select a subsystem from the list and then modify the values on the lower portion of the window. When the new values have been entered, select **Update** to commit the changes. To remove the overridden threshold values, select the **Set To Defaults** button. The changes will be committed and displayed in the Subsystem Thresholds table.

After any modifications to the subsystem-specific thresholds, the Core Server and Migration Purge Server for those subsystems must be recycled to make the changes effective.

Field Descriptions

Storage Class ID. The ID of the storage class to which these subsystem-specific thresholds apply.

Storage Class Type. The type of storage class (disk or tape).

Storage Class Name. The Descriptive Name of the storage class.

Default Warning Threshold. The default warning threshold for this storage class expressed as a percentage of used space.

Default Critical Threshold. The default critical threshold for this storage class expressed as a percentage of used space.

Storage Subsystem Count. The number of storage subsystems in the HPSS system.

Subsystem List. A list of configured storage subsystems. Columns in the list include:

Subsys ID. The ID of the selected storage subsystem.

Subsys Name. The name of the selected storage subsystem.

Warning. The current warning threshold value for the selected subsystem. If the storage class defaults are to be used, the text "default" will be displayed.

Critical. The current critical threshold value for the selected subsystem. If the storage class defaults are to be used, the text "default" will be displayed.

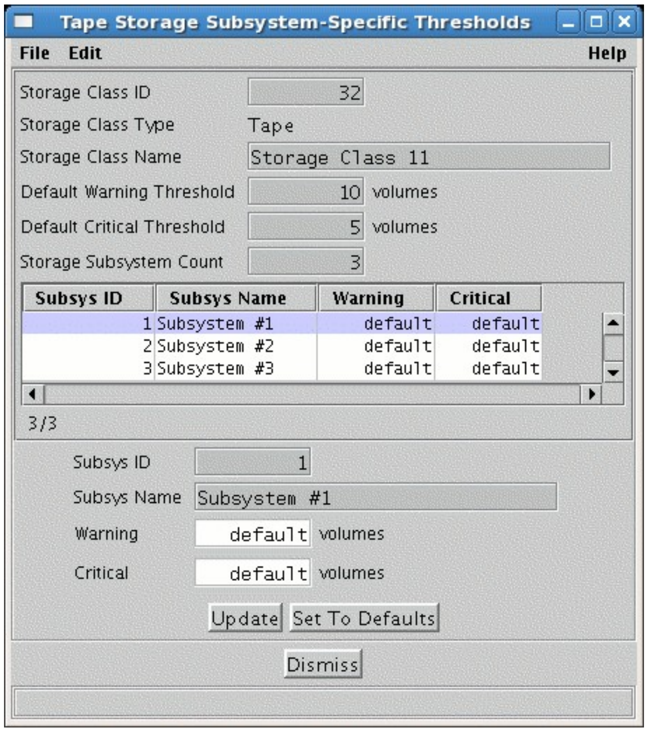
Buttons

Set To Defaults. A button to remove the threshold override values from the selected storage subsystem. Select a storage subsystem row from the Subsystem List. The selected storage subsystem information will be displayed in the lower panel. Pressing the **Set To Defaults** button will commit the changes to the selected subsystem and "default" will be displayed for both the warning and critical thresholds.

Related Information

Section 4.2.3.8: *Storage Class Threshold Overrides* on page 81

6.1.4.2. Tape Storage Subsystem-Specific Thresholds



Storage Class ID	32
Storage Class Type	Tape
Storage Class Name	Storage Class 11
Default Warning Threshold	10 volumes
Default Critical Threshold	5 volumes
Storage Subsystem Count	3

Subsys ID	Subsys Name	Warning	Critical
1	Subsystem #1	default	default
2	Subsystem #2	default	default
3	Subsystem #3	default	default

3/3

Subsys ID	1
Subsys Name	Subsystem #1
Warning	default volumes
Critical	default volumes

This window is used to define **Warning** and **Critical** thresholds unique to a particular storage subsystem, overriding the values defined in the tape storage class. The user may modify either the number of

Warning volumes, **Critical** volumes or both for one or more of the listed subsystems. Select a subsystem from the list and then modify the values on the lower portion of the window. When the new values have been entered, select **Update** to commit the changes. To remove the customized threshold values, select the desired subsystem and click the **Set To Defaults** button. The changes will be committed and displayed in the Subsystem Thresholds table.

After any modifications to the thresholds, the Core Server and Migration Purge Server for the affected subsystem must be recycled to make the changes effective.

Field Descriptions

Storage Class ID. The ID of the storage class to which these subsystem-specific thresholds apply.

Storage Class Type. The type of storage class (disk or tape).

Storage Class Name. The Descriptive Name of the storage class.

Default Warning Threshold. The default warning threshold for this storage class expressed as a number of free volumes.

Default Critical Threshold. The default critical threshold for this storage class expressed as a number of free volumes.

Storage Subsystem Count. The number of storage subsystems in the HPSS system.

Subsystem List. A list of configured storage subsystems. Columns in the list include:

> **Subsys ID.** The ID of the selected storage subsystem.
>
> **Subsys Name.** The name of the selected storage subsystem.
>
> **Warning.** The current warning threshold value for the selected subsystem. If the storage class defaults are to be used, the text “default” will be displayed.
>
> **Critical.** The current critical threshold value for the selected subsystem. If the storage class defaults are to be used, the text “default” will be displayed.

Buttons

Set To Defaults. A button to remove the threshold override values from the selected storage subsystem. Select a storage subsystem row from the Subsystem List. The selected storage subsystem information will be displayed in the lower panel. Pressing the **Set To Defaults** button will commit the changes to the selected subsystem and “default” will be displayed for both the warning and critical thresholds.

Related Information

Section 4.2.3.8: *Storage Class Threshold Overrides* on page 81

6.1.5. Changing a Storage Class Definition

A storage class definition can be changed by bringing up the *Storage Class Configuration* window and making the desired modifications.

Fields that can be changed in the storage class definition without major impact are the **Migration Policy**, **Purge Policy**, **Warning Threshold**, and **Critical Threshold**. They can be changed at the discretion of the administrator to reflect desired behavior.

If files have been stored in a storage class without a migration policy, and a migration policy is subsequently configured for it, the files created before the addition of the policy will not be migrated. Use the **mkmprec** utility to create migration records for these files so that they will migrate properly. See the **mkmprec** man page for more information.

When a migration or purge policy is modified, or when a storage class configuration is modified to use a different migration policy, the affected Migration Purge Servers and Core Servers must be restarted in order for the new policy to take effect.

Optimum Access Size, **Device I/O Rate (DIOR)**, and **Average Latency** are used only in an advisory capacity by HPSS unless a site has written a special interface using the Client API that takes advantage of the COS Hints capability and uses these fields in doing so. Generally, they can be changed at will, but should be set to accurately reflect their intended usage.

Maximum VVs To Write can be changed as desired. Changes made to this field will not take effect until the Core Server(s) providing service to the storage class are re-started.

Max Multiplier (MAXMULT), **Min Multiplier (MINMULT),** and **Average Number of Storage Segments** can be changed as desired for disk storage classes.

To change the **Storage Segment Size** fields **Media Type**, **Stripe Width (SW)**, or **Blocks Between Tape Marks (BBTM)**, the existing storage class definition must be deleted and recreated with the new desired values. Take care to either remove migration and purge policies that are no longer needed or reassign them to the revised storage class. Because a number of servers cache storage class definitions, it is best to make these changes when only SSM is running. Care should be taken in making these changes. See the *HPSS Installation Guide*, Section 3.10: *Storage Characteristics Considerations*.

6.1.6. Deleting a Storage Class Definition

Before deleting a storage class definition, be sure that all of the subsystem-specific warning and critical thresholds are set to “default”. If this is not done, one or more threshold records will remain in metadata and will become orphaned when the storage class definition is deleted.

To delete a storage class definition, ensure that no files exist in this storage class and it is no longer referenced in any hierarchy. All of the virtual volumes in this storage class should also be deleted, using the Delete Resources or Reclaim operation. Once the storage class is empty and it is not referenced from any hierarchy definition, the storage class definition can be removed.

See Section 8.7: *New Storage Technology Insertion* on page 291 for related information.

See also the **dump_acct_sum** utility, which can quickly verify the number of files in each COS.

It is recommended HPSS Customer Support be called for assistance with this operation.

6.2. Storage Hierarchies

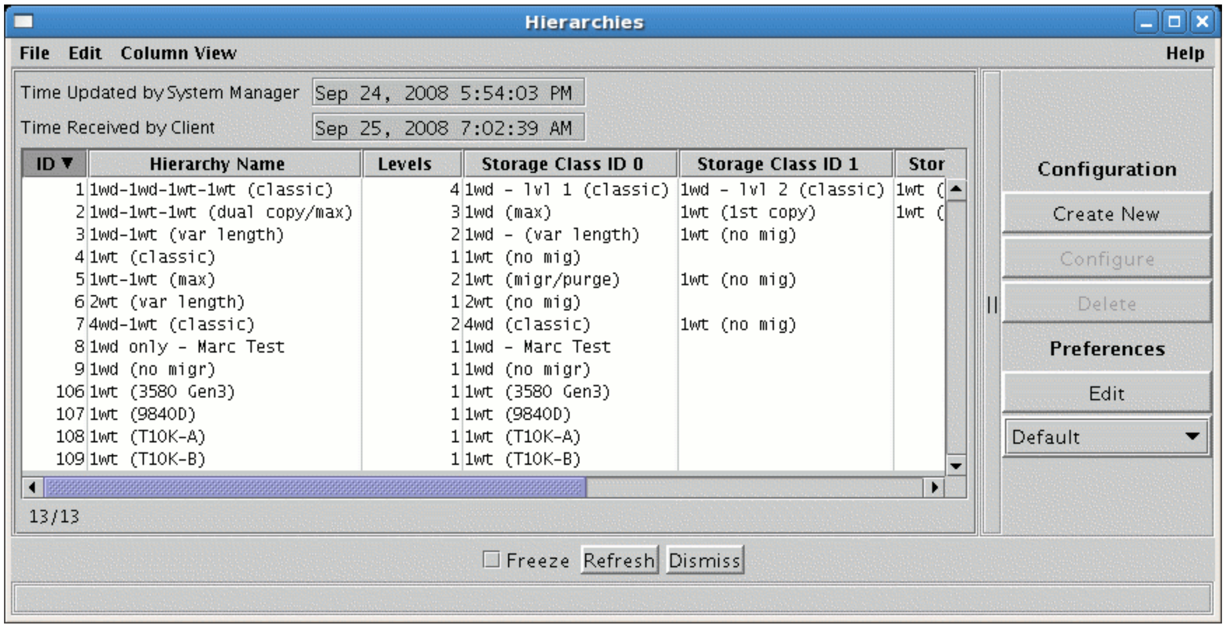
This section describes how to create, modify, and delete storage hierarchies.

6.2.1. Hierarchies Window

An HPSS storage hierarchy can be created and managed using the *Hierarchies* window. This window is accessed from the *HPSS Health and Status* window's **Configure** menu, submenu **Storage Space**, item **Heirarchies.** Refer to Section 3.9.3: *HPSS Health and Status* on page 58.

The following rules for creating storage hierarchies are enforced by the *Hierarchies* window:

- A storage class may be used only once per hierarchy.
- Disk migration may migrate to one or more target levels in the hierarchy. To create multiple copies of data, select multiple migration targets.
- Tape migration may only migrate to a single target level in the hierarchy.
- A given storage class may be a migration target for only one storage class above it in the hierarchy (two different storage classes in the hierarchy may not both migrate to the same storage class).



Time Updated by System Manager: Sep 24, 2008 5:54:03 PM

Time Received by Client: Sep 25, 2008 7:02:39 AM

ID ▼	Hierarchy Name	Levels	Storage Class ID 0	Storage Class ID 1	Stor
1	1wd-1wd-1wt-1wt (classic)	4	1wd - lvl 1 (classic)	1wd - lvl 2 (classic)	1wt (
2	1wd-1wt-1wt (dual copy/max)	3	1wd (max)	1wt (1st copy)	1wt (
3	1wd-1wt (var length)	2	1wd - (var length)	1wt (no mig)	
4	1wt (classic)	1	1wt (no mig)		
5	1wt-1wt (max)	2	1wt (migr/purge)	1wt (no mig)	
6	2wt (var length)	1	2wt (no mig)		
7	4wd-1wt (classic)	2	4wd (classic)	1wt (no mig)	
8	1wd only - Marc Test	1	1wd - Marc Test		
9	1wd (no migr)	1	1wd (no migr)		
106	1wt (3580 Gen3)	1	1wt (3580 Gen3)		
107	1wt (9840D)	1	1wt (9840D)		
108	1wt (T10K-A)	1	1wt (T10K-A)		
109	1wt (T10K-B)	1	1wt (T10K-B)		

13/13

This window allows you to manage storage hierarchies. It lists all currently defined hierarchies and allows you to change existing hierarchies. New hierarchies can also be created from this window.

Field Descriptions

Hierarchy List columns

ID. The unique ID assigned to the storage hierarchy when it was configured.

Hierarchy Name. The descriptive name of the hierarchy.

Levels. The number of storage classes that have been configured into the hierarchy.

Storage Class ID 0 to **Storage Class ID 4.** The name of the storage class at the given hierarchy level. HPSS arranges the storage classes into levels internally.

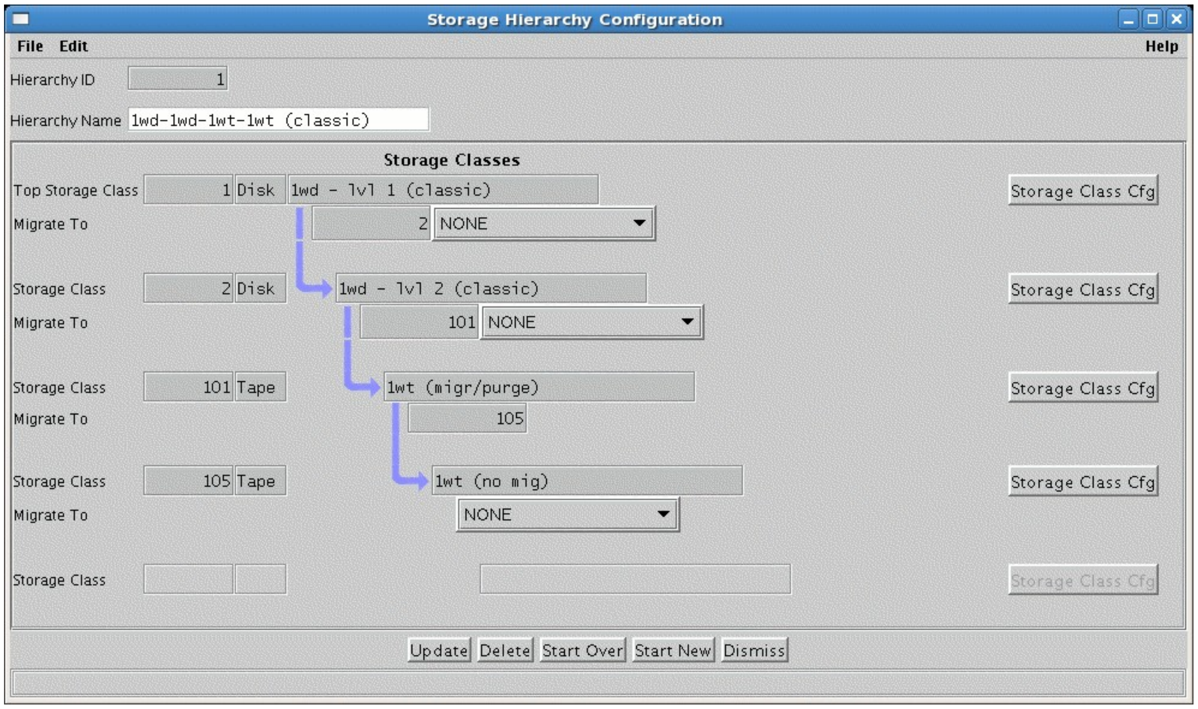
Configuration Buttons

Create New. Open a *Storage Hierarchy* window with default values.

Configure. Open the selected storage hierarchy configuration for editing. One hierarchy from the list must be selected before this button is active.

Delete. Delete the selected storage hierarchy(s).

6.2.2. Storage Hierarchy Configuration Window



Hierarchy ID: 1

Hierarchy Name: 1wd-1wd-1wt-1wt (classic)

Top Storage Class	1	Disk	1wd - lvl 1 (classic)	
Migrate To			2	NONE
Storage Class	2	Disk	1wd - lvl 2 (classic)	
Migrate To			101	NONE
Storage Class	101	Tape	1wt (migr/purge)	
Migrate To			105	
Storage Class	105	Tape	1wt (no mig)	
Migrate To				NONE
Storage Class				

This window allows an administrator to manage a storage hierarchy.

A maximum of 5 storage classes can be configured into a hierarchy. A single storage class may appear in multiple hierarchies. *No hierarchy can reference the same storage class more than once.*

Note that in addition to defining the hierarchy, the appropriate migration policies must be configured for each storage class in the hierarchy. Without the proper migration policies, neither migration nor duplicate tape copies will be maintained. See Section 6.4 *Migration Policies* on page 180.

Once a storage hierarchy has been created, it should not be deleted as long as there is a Class of Service entry associated with it. A Class of Service entry should never be deleted unless all files in that Class of Service have been deleted and no client is still attempting to use that Class of Service.

After changing any hierarchy information, all Core Servers and Migration Purge (MPS) servers must be recycled to make the changes effective.

Field Descriptions

Hierarchy ID. The ID associated with this hierarchy . Any unique, positive 32-bit integer value. The default value is the last configured ID plus 1.

Hierarchy Name. The descriptive name associated with this hierarchy. The default value is "Hierarchy <ID>".

Top Storage Class. The storage class at the highest level of the hierarchy.

After a storage class is selected, a new storage class drop-down list will appear in the **Migrate To** field. Selecting a storage class from the drop-down list will define the storage class to which the **Top Storage Class** will migrate, and an arrow will show the migration path from the **Top Storage Class** to the second Storage Class. .

This process may be repeated to create more migration paths.

Depending on whether the **Top Storage Class** is disk or tape, either one or two new storage class drop-down lists will appear.

Migrate To. The storage class id(s) to which the a Storage Class will migrate. Initially, when creating a new hierarchy, these fields will not be visible. As the user configures the hierarchy, the **MigrateTo** values will be displayed. After a storage class has been selected from a drop-down list, the drop-down list will be replaced with a non-editable storage class id corresponding to the user's selection. One or more new storage class drop-down lists will become visible.

Storage Class. The id number of the storage class at the second through the fifth level of the hierarchy. See description above for **Top Storage Class**.

Storage Class Cfg button. Clicking this button will open the *Storage Class Configuration* window corresponding to the storage class name to the left of the button.

6.2.3. Changing a Storage Hierarchy Definition

A storage hierarchy definition can be changed by bringing up the *Storage Hierarchy Configuration* window and making the desired modifications. Note that the only modification that is possible to an existing storage hierarchy is the inclusion of additional storage levels at the bottom of the hierarchy. All other modifications require that the hierarchy be deleted and recreated.

When adding a migration target to the bottom of a storage hierarchy, be sure that the migrating storage class is configured with appropriate migration and purge policies. Without migration and purge policies, no migration and purge will take place even if target levels exist in the hierarchy. Furthermore, any pre-existing data in the storage class (data which was written into the storage class before the storage class was configured to use migration and purge policies) will not be migrated and purged even after the migration and purge policies are added. The migration and purge policies are honored only for data written after the policies are put into effect. To migrate and purge data written before these policies became effective, use the **mkmprec** utility.

6.2.4. Deleting a Storage Hierarchy Definition

The rules that apply for deleting a Class of Service also apply for deleting a storage hierarchy because in most systems a one-to-one relationship exists between hierarchies and Classes of Service. Before deleting a hierarchy, all the files in all of the Classes of Service that use the hierarchy must be deleted. Failure to do this will render the files unreadable.

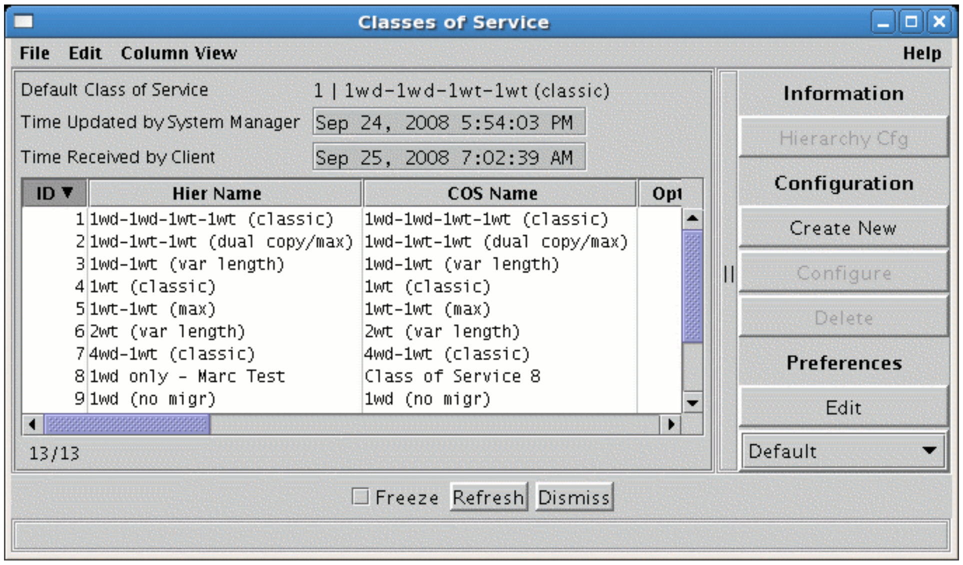
It is recommended that HPSS Customer Support be called to assist with this operation.

6.3. Classes of Service

This section describes how to configure classes of service.

6.3.1. Classes of Service Window

A COS can be created and managed using the *Classes of Service* window.



Default Class of Service 1 | 1wd-1wd-1wt-1wt (classic)
Time Updated by System Manager Sep 24, 2008 5:54:03 PM
Time Received by Client Sep 25, 2008 7:02:39 AM

ID	Hier Name	COS Name	Opt
1	1wd-1wd-1wt-1wt (classic)	1wd-1wd-1wt-1wt (classic)	
2	1wd-1wt-1wt (dual copy/max)	1wd-1wt-1wt (dual copy/max)	
3	1wd-1wt (var length)	1wd-1wt (var length)	
4	1wt (classic)	1wt (classic)	
5	1wt-1wt (max)	1wt-1wt (max)	
6	2wt (var length)	2wt (var length)	
7	4wd-1wt (classic)	4wd-1wt (classic)	
8	1wd only - Marc Test	Class of Service 8	
9	1wd (no migr)	1wd (no migr)	

13/13

This window lists the classes of service that are currently configured. It also allows an administrator to update and delete existing classes of service and to add a new class of service. This window is displayed by selecting the Configure menu from the *HPSS Health and Status* window, then selecting the Storage Space submenu, and then selecting the Classes of Service submenu item.

Field Description

Default Class of Service. The default class of service. This is displayed for information only and may not be changed from this window. It may be changed from the *Global Configuration* window.

Class of Service List. The central panel in this window shows the list of configured classes of service. The columns display the fields from the *Class of Service Configuration* window discussed in the following section.

Information Buttons.

Hierarchy Cfg. Open a *Storage Hierarchy Configuration* window for the hierarchy associated with the selected class of service.

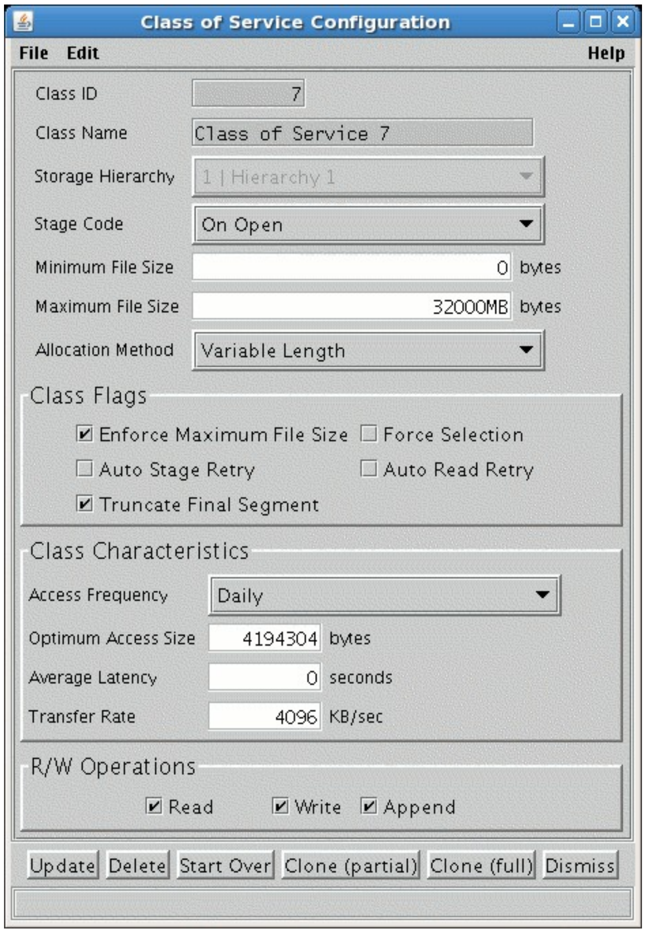
Configuration Buttons.

Create New. Open a *Class of Service* window containing default values for a new class of service.

Configure. Open the selected class(es) of service configuration(s) for editing.

Delete. Delete the selected class(es) of service.

6.3.2. Class of Service Configuration Window


Class of Service Configuration
File Edit Help
Update Delete Start Over Clone (partial) Clone (full) Dismiss

Class ID	7
Class Name	Class of Service 7
Storage Hierarchy	1 | Hierarchy 1
Stage Code	On Open
Minimum File Size	0 bytes
Maximum File Size	32000MB bytes
Allocation Method	Variable Length

Class Flags

- ☑ Enforce Maximum File Size
- ☐ Force Selection
- ☐ Auto Stage Retry
- ☐ Auto Read Retry
- ☑ Truncate Final Segment

Class Characteristics

Access Frequency	Daily
Optimum Access Size	4194304 bytes
Average Latency	0 seconds
Transfer Rate	4096 KB/sec

R/W Operations

☑ Read ☑ Write ☑ Append

This window allows an administrator to manage a class of service.

Field Descriptions

Class ID. The unique integer ID for the COS. Any positive 32-bit integer value.

Class Name. The descriptive name of the COS. A character string up to 31 bytes in length. The default value is "Class of Service <ID>".

Storage Hierarchy. The name of the storage hierarchy associated with this COS.

Stage Code. A code that indicates the file staging options. Valid values are On Open, On Open Async, No Stage, and On Open Background. For the first COS created in a new HPSS system, the default value is On Open. For all subsequently created COSes, the default value is the same as the most recent COS configured.

Advice – Changing the Stage Code should be done with care. See Section 6.3.3: Changing a Class of Service Definition on page 178 for detailed information on each of the choices for Stage Code.

Minimum File Size. The size, in bytes, of the smallest bitfile supported by this COS. Valid values are any positive 64-bit integer value.

Maximum File Size. The size, in bytes, of the largest bitfile supported by this COS. Valid values are any positive 64-bit integer value.

Allocation Method. How disk storage segments will be allocated in the storage class at the top level of the hierarchy used by this COS. Ignored if the top level storage class is tape. Three allocation methods are available:

- **Fixed Length, Classic Style.** All storage segments allocated for a single file, except possibly the last segment of the file, are the same size. This size is computed at file creation time to be the size which wastes the least disk space, based on the file size, the minimum and maximum segment sizes configured for the storage class, and the average number of segments per file configured for the storage class. In order to take maximum advantage of this allocation method, the application creating the file must specify the file size in its file creation hints.

  The application can force this fixed segment size to be the maximum segment size configured for the storage class by specifying HINTS_FORCE_MAX_SSEG in the file creation hints.

  If the application does not force the segment size to be the max size and it does not provide hints telling the core server the file size, then the core server will use the minimum segment size configured for the storage class.

  If the file is purged completely from the top level and later restaged to the top level, a new segment size will be computed at stage time based on the current file size.

- **Fixed Length, Max Style.** The size of each storage segment allocated for a file, except possibly the last segment, is the maximum segment size configured for the storage class.

- **Variable Length.** The size of the first storage segment allocated for a file is the minimum segment size configured for the storage class. The size of each successive storage segment is twice that of the preceding segment, up to the maximum segment size configured for the storage class. Once the max segment size is reached, each remaining segment except possibly the last is also the max segment size.

Under all three allocation methods, the last segment of the file may be truncated to a size to best fit the actual data contained in the segment. See the **Truncate Final Segmen**t option in the **Class Flags** section below.

Class Flags

- **Enforce Maximum File Size.** When set, a bitfile larger than the Maximum File Size cannot be created in this COS.
- **Force Selection.** A flag to determine how a COS will be selected. If ON, a client must explicitly select this COS in order to have a file assigned to it; if the client merely supplies general COS hints for a file, this COS will not be selected.

- **Auto Stage Retry**. When this flag is turned on, and a valid secondary copy of the data exists, and a stage from the primary copy fails, HPSS will automatically retry the stage using the secondary copy.
- **Auto Read Retry.** When this flag is turned on, and a valid secondary copy of the data exists, and an attempt to read the first copy fails, HPSS will automatically retry the read using the secondary copy. This retry will not be attempted if any data was transferred between the client and HPSS during the failure.

  For **Auto Stage Retry** and **Auto Read Retry** to work properly, the COS must contain at least two copies of the files and at least one valid second copy must have been created during HPSS migration processing. If any tape storage class in the COS has a migration policy, that policy must be configured for **Migrate Files** or **Migrate Files and Purge.** A storage class whose migration policy is configured for **Migrate Volumes** or **Migrate Volumes and Whole Files** will not be considered as a valid alternate copy from which to retry a failed read or stage operation. Any storage class to which such a storage class migrates, directly or indirectly, will not be considered as a valid retry candidate, either.
- **Truncate Final Segment**. A flag to influence whether the final segment of files will be truncated at file close time in order to save disk space.

  When a file is closed, its final segment may be truncated to the smallest segment size which is valid for the storage class and will still hold the actual data written to the segment. This size will be a power-of-two multiple of the minimum segment size configured for the storage class. The segment will later be expanded to its original size if the file is re-opened and new data is written to the segment or to a point in the file beyond the segment. Truncation of the final segment is performed only on the file data at the top level of the storage hierarchy and only if the storage class at the top level is disk.

  Truncation of the final segment is controlled by the NOTRUNC_FINAL_SEG flag in the bitfile descriptor; if this flag is off, truncation is performed at file close, and if it is on, truncation is not performed.

  The COS **Truncate Final Segment** flag influences whether the NOTRUNC_FINAL_SEG flag in the bitfile descriptor is set at file creation time. The user may specify a creation hint to turn truncation off even if the COS allows truncation, but he may not turn truncation on if the COS prohibits it. If the COS **Truncate Final Segment** flag is on, then at file creation time the NOTRUNC_FINAL_SEG flag will be set to off, unless the user specifies the NOTRUNC_FINAL_SEG hint. If the COS **Truncate Final Segment** flag is off, then the NOTRUNC_FINAL_SEG flag will be set to on at file creation time, and the user may not override this.

  Users may also turn on the NOTRUNC_FINAL_SEG flag in their own files after file creation by use of the hpss_SetFileNotrunc client api function, but they may not turn off the flag. Authorized callers may use the hpss_SetFileNotrunc function to turn the flag on or off for any file. Authorized callers are those such as hpssssm who have CONTROL permission on the core server's Client Interface ACL.

  Advice – Truncation of the final segment normally saves disk space. However, if a file is frequently written, closed, re-opened, and appended to, the repeated truncation and re-expansion of the final segment could result in fragmentation of the disk, as the space for the expansion is not guaranteed to be adjacent to the original segment. In addition, it is possible that the only extent available for the expansion is larger than needed, so some space could actually be wasted.

  If it is known that the users of a COS do frequent appends to their files, it is better to turn the COS ***Truncate Final Segment*** *flag off to avoid these potential fragmentation and space waste issues. If no* *users do frequent appends, or if those who do can be relied upon to turn truncation off for their own files, or if the system administrator can easily identify files which are frequently appended and can turn off truncation on them individually, then the site might want to take advantage of the space savings for the remaining files and leave* ***Truncate Final Segment*** *on in the COS definition.*

One additional consideration is that truncating the final segment incurs a small performance penalty. In development testing, this penalty was measured at 14% for writing 100 files with truncation in comparison to writing them without truncation; this value may vary under different site configurations.

Class Characteristics

- **Access Frequency.** The frequency, on average, for accessing files in this COS. Valid values are Hourly, Daily, Weekly, Monthly or Archive.
- **Optimum Access Size.** The suggested number of bytes that should be written at one time for maximum efficiency. (Not currently used by HPSS. May be used by site specific applications)
- **Average Latency.** The average time, in seconds, that elapses between the time a transfer request is accepted for processing and the time the data transfer begins.
- **Transfer Rate**. The average throughput (in KB per second) that can be transferred using this COS.

R/W Operations

- **Read**. If ON, files in this COS are readable, subject to the permissions on each individual file.
- **Write**. If ON, files in this COS are writable, subject to the permissions on each individual file.
- **Append**. If ON, files in this COS can have new data appended to them, subject to the permissions on each individual file. This button may only be ON if the **Write** button is also ON. If the **Write** button is OFF, the **Append** button is disabled.

  In all cases, whether an individual file can be read or written will be determined by the permissions on that particular file. It may be the case that a COS is RW, but certain files within that COS may have their permissions set such that they can be neither read nor written.

6.3.3. Changing a Class of Service Definition

A COS definition can be changed by bringing up the *Class of Service Configuration* window and making the desired modifications. Fields on the *Class of Service Configuration* window that can be changed without major impact are the Access Frequency, Optimum Access Size, Average Latency, and Transfer Rate. These fields are advisory in nature and are currently used only by the Client API. Users of custom HPSS interfaces will have to assess the impact of any change.

The **Maximum File Size** can be changed, but care should be exercised when doing so. Increasing this value may result in storing files that are inappropriate for the hierarchy supporting this COS. For example, suppose a hierarchy is defined with disk at the top level with a storage segment size of 64 KB and the **Maximum File Size** in this COS is set to 1 MB. Changing the maximum file size to 2 GB in this class would be inappropriate because 32768 64KB storage segments would be needed to contain such a file.

Associated with the **Maximum File Size** is the **Enforce Maximum File Size** flag. Changing this can have a significant impact. Turning the flag on constrains files that are already larger than the **Maximum File Size** to their current size. Existing smaller files will be constrained to the **Maximum File Size**.

Changing **Minimum File Size** can have an impact on COS selection. Currently, the PFTP and FTP interfaces use the Minimum File Size to select an appropriate COS based on file size.

Changing the **Stage Code** should be done with care:

- The **On Open** option stages files to the top of the hierarchy when the file is opened, and is synchronous. The open operation returns when the stage operation completes.
- The **No Stage** option can have a significant impact on system performance. Repeated reads of files that have been migrated will generally be satisfied from lower levels in the hierarchy where files are likely to be on tape. If users are writing only parts of a file at a time with significant delays between the writes, migration may result in the file being stored in a more fragmented manner on tape.
- The **On Open Async** option causes stage operations to be started when files are opened, but open operations do not wait for stages to complete; they return to their callers immediately. If a caller then reads one of these files, the read operation will block until the stage is complete.
- The **On Open Background** option causes an open operation to complete immediately, returning a special error code that indicates that the stage is not complete. The caller can then poll HPSS for completion of the stage operation before reading the file. Normally this polling is done automatically by the client API, but client API polling can be turned off which will allow the client to do the polling.

Changing the **Auto Stage Retry** flag has no adverse side effects. It is recommended that this flag be enabled for all COS's using multiple copies.

The hierarchy associated with a COS cannot be changed without deleting all files in this COS or moving them to another COS. Failure to observe this condition will likely result in lost and corrupted files. Any changes to an existing COS other than changing the hierarchy ID can be put into effect by re-initializing core servers that use the COS. If a COS is deleted or added, the core servers must be recycled to recognize this.

6.3.4. Deleting a Class of Service Definition

To delete a COS, you must ensure that all files in the COS have been either deleted or moved to another COS and that all configuration references to the COS have been removed. These configurations include the Global Configuration's Default Class of Service, Storage Subsystem's Default COS Override, Log Daemon's Archive Class of Service, and every fileset's associated Class of Service.

Deleting a Class of Service is an unusual administrative action and HPSS does not provide utility programs to delete all the files in a given COS or move all files in a given COS to another COS. A certain amount of site specific ad hoc planning and usage of available tools will be necessary to identify, remove or move all files from a Class of Service. The **dump_acct_sum** utility program provides a count of the number of files in all Classes of Service based on the accounting metadata and was developed primarily so there would be a reasonable way to know if a Class of Service was empty.

To delete the COS definition, use the **Delete** button on the appropriate *Class of Service Configuration* window.

6.3.5. Changing a File's Class of Service

The Core Server provides a means to change the class of service of a file. The Core Server moves the body of the file as appropriate to media in the destination Class of Service, then allows the usual migration and purge algorithms for the new Class of Service to apply. The file body is removed from the media in the old Class of Service. If large numbers of files on tapes are changed, an opportunity to retire, repack and/or reclaim tapes in the old Class of Service may arise.

The scrub utility can be used to initiate the COS change request of a file as follows:

```
% scrub
scrub> changecos <fullPathname> <newCOSid>
scrub> quit
```

6.3.6. Canceling a Class of Service Change Request

A Class of Service change request can be canceled in order to deal with a persistent problem in changing the file.

The scrub utility can be used to cancel the COS change request as follows:

```
% scrub
scrub> changecos <fullPathname> 0
scrub> quit
```

The associated Core Server COS change record will be removed immediately. However, this request may fail if the existing COS change request is currently being processed by one of the Core Server's COS change threads. In this case, the user should attempt to reissue the request later.

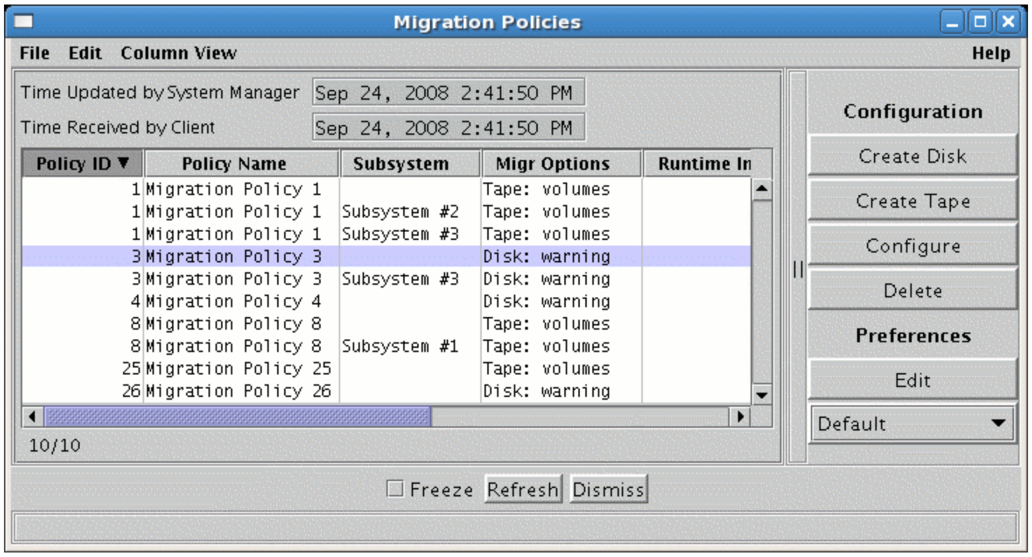
6.4. Migration Policies

A migration policy is associated with a storage class and defines the criteria by which data is migrated from that storage class to storage classes at lower levels in the storage hierarchies. Note, however, that it is the storage hierarchy definitions, not the migration policy, which determine the number and location of the migration targets. Also, note that a storage class may be used in multiple hierarchies and may have different migration targets in each.

A basic migration policy must be assigned to any disk or tape storage class that requires migration services. This basic policy determines the migration parameters for the storage class across all storage subsystems. Storage subsystem specific migration policies may be added which override the basic policy in the selected subsystems. The storage subsystem specific migration policies share the migration Policy ID and Policy Name with the basic policy.

6.4.1. Migration Policies Window

This window displays a list of all the configured migration policies.



Time Updated by System Manager: Sep 24, 2008 2:41:50 PM
Time Received by Client: Sep 24, 2008 2:41:50 PM

Policy ID	Policy Name	Subsystem	Migr Options	Runtime In
1	Migration Policy 1		Tape: volumes	
1	Migration Policy 1	Subsystem #2	Tape: volumes	
1	Migration Policy 1	Subsystem #3	Tape: volumes	
3	Migration Policy 3		Disk: warning	
3	Migration Policy 3	Subsystem #3	Disk: warning	
4	Migration Policy 4		Disk: warning	
8	Migration Policy 8		Tape: volumes	
8	Migration Policy 8	Subsystem #1	Tape: volumes	
25	Migration Policy 25		Tape: volumes	
26	Migration Policy 26		Disk: warning	

10/10

Both basic and subsystem specific migration policies are created and managed using the *Migration Policies* window. The basic policy must be created before creating any subsystem specific policies.

The fields in the basic policy are displayed with default values. Change any fields to desired values as needed. Click on the **Add** button to write the new basic policy to the HPSS metadata.

To configure a subsystem specific policy, select an existing basic policy and click on Configure. The basic policy will be displayed in a tabbed window with a tab for each storage subsystem for which subsystem specific thresholds have been created. Click on the tab for the desired storage subsystem and make the necessary subsystem specific changes, and then click on the **Update** button. To add a specific threshold configuration for a new subsystem, click the **New Subsystem Policy** button.

When a migration policy is added to an existing storage class, the Migration Purge Servers for that storage class must be restarted in order for the policy to take effect.

When HPSS needs to apply a migration policy to a subsystem, it first searches for a matching subsystem policy. If none is found, HPSS uses the basic policy. You can consider the basic policy to be the default for all subsystems which do not have policies of their own.

If a policy has one or more subsystem-specific overrides, the basic and subsystem-specific policies will be listed separately. Each entry will have the same ID, but the **Subsystem** field will be blank for the basic policy and contain the subsystem ID for each subsystem-specific override.

After changing any migration policy information, the changes will take effect for each Migration Purge Server (MPS) when the MPS is restarted, or when it rereads the changed policy.

Field Descriptions

Policy ID. The unique ID number assigned to this migration policy.

Policy Name. The name assigned to this migration policy.

Subsystem. If the policy is a subsystem-specific override, this shows the subsystem to which it applies. This field will be blank for basic policies.

Other Migration Policy List columns. The remaining columns provide the same information that can be found in Section 6.4.2.1: *Disk Migration Policy Configuration* on page 182 and Section 6.4.2.2: *Tape Migration Policy Configuration* on page 185 windows.

Configuration Buttons.

Create Disk. Opens a *Disk Migration Policy* window with default values.

Create Tape. Opens a *Tape Migration Policy* window with default values.

Configure. Opens the selected migration policy for editing.

Delete. Deletes the selected migration policy(s).

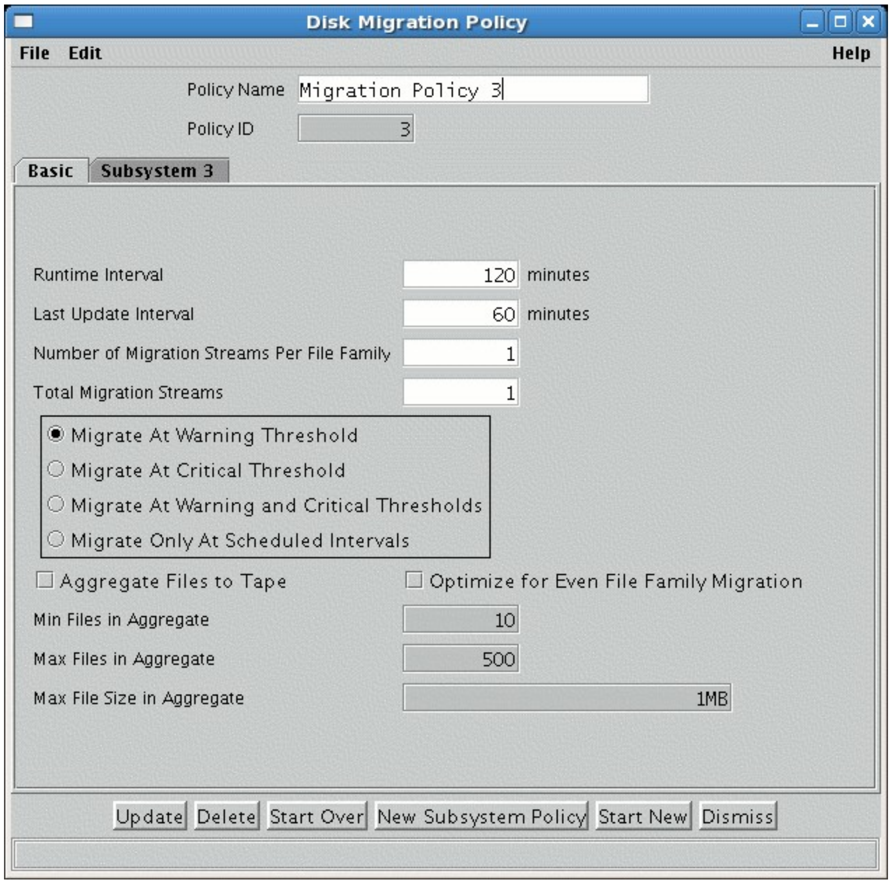
Related Information

HPSS Installation Guide, Section 3.7.2: *Migration Purge Server*

6.4.2. Migration Policy Configuration

This section describes the configuration, update, and deletion of migration policies.

6.4.2.1. Disk Migration Policy Configuration



Policy Name	Migration Policy 3
Policy ID	3
Runtime Interval	120 minutes
Last Update Interval	60 minutes
Number of Migration Streams Per File Family	1
Total Migration Streams	1

● Migrate At Warning Threshold
○ Migrate At Critical Threshold
○ Migrate At Warning and Critical Thresholds
○ Migrate Only At Scheduled Intervals

☐ Aggregate Files to Tape
☐ Optimize for Even File Family Migration

Min Files in Aggregate	10
Max Files in Aggregate	500
Max File Size in Aggregate	1MB

This window allows an administrator to manage disk migration policies and their subsystem-specific overrides.

Subsystem-specific policies define migration rules to be applied on a subsystem basis instead of using the default migration policy. When a migration run occurs and the storage class's migration policy does not have a subsystem-specific migration policy with the same subsystem ID, then the basic migration policy will be applied; otherwise, the matching subsystem-specific migration policy will be used. Click the **Create Disk** button on the *Migration Policies* window to create a new policy. To bring up the configuration for an existing policy, select the policy from the list and click on the **Configure** button.

Field Descriptions

Policy Name. The descriptive name of the migration policy.

Policy ID. A unique ID associated with the migration policy.

Runtime Interval. The number of minutes to wait between the end of one migration run and the start of the next. This can be any positive value up to 1000000.

*Advice - The **Runtime Interval** is the maximum number of minutes from the end of one migration run to the start of the next. Migration needs to run often enough so the storage class to which the policy is assigned does not fill up. Migrations can also be started manually from the Active HPSS Storage Classes window.*

Last Update Interval. The number of minutes that must pass since a file was last updated before it can become a candidate for migration.

Number of Migration Streams Per File Family. The number of migration streams to be allocated to each file family. This value effectively determines how many file families can be migrated simultaneously. For example: **Number of Migration Streams Per File Family** is set to 2 and **Total Migration Streams** is set to 10; up to five families will be migrated simultaneously (five families with two streams working on each family yields a total of 10 migration streams).

Total Migration Streams. The number of parallel migration threads which will be run during migration for each copy level in the hierarchy. This value must be a multiple of the *Number of Migration Streams Per File Family*.

*Advice - Setting the **Total Migration Streams** value too high may result in higher than desired tape drive utilization. Each hierarchy within a disk storage class is migrated in a separate but parallel fashion. As such, when there are files eligible for migration in each hierarchy within a disk storage class, tape mount requests will be generated for each hierarchy. The **Total Migration Streams** value may be used indirectly to control how many tape mounts are requested, and thus how many drives are used during migration. The maximum number of drives needed for a migration can be calculated as the stripe width value (**SW**) multiplied by the **Total Migration Streams** value (**TMS**) for every hierarchy and disk-to-tape copy level within the storage class:*

$$(SW \times TMS)_{Hier1,LevelA} + (SW \times TMS)_{Hier1,LevelB} + \cdots$$

$$(SW \times TMS)_{Hier2,LevelA} + (SW \times TMS)_{Hier2,LevelB} + \cdots$$

$$+ \cdots +$$

$$(SW \times TMS)_{Hiern,Levelx} + (SW \times TMS)_{Hiern,Levely} + \cdots$$

For example, an HPSS system that has a disk storage class which migrates to a tape storage class with stripe width two. The migration policy's request count is also two. In this situation, the maximum tape drive usage would be four: two sets of two-wide tape VVs.

A hierarchy that migrates data from disk to a first and second 4-wide tape copy with a request count of one would use a maximum of eight tape drives: the first copy would go to one set of 4-wide tapes and the second copy would migrate to a second set of 4-wide tapes. A second hierarchy added to the mix with the same characteristics would double the number of tape drives needed.

Now consider an HPSS system that has disk Storage Class A which contains Hierarchy 1 and Hierarchy 2. Hierarchy 1 migrates to a tape storage class with stripe width of two and has a request count of four. Hierarchy 2 migrates to a first copy tape storage class with stripe width of two and a request count of three, and a second copy tape storage class with a stripe width of one and a request count of two. The total drive utilization for a maximal migration case (when there exist files eligible for migration in all hierarchies of the disk storage class) will be sixteen:

$$(2_{SW} \times 4_{RC})_{Hier1,Level1} + (2_{SW} \times 3_{RC})_{Hier2,Level1} + (1_{SW} \times 2_{RC})_{Hier2,Level2} = 16$$

In addition to the Request Count, the Maximum VVs to Write *parameter in the* Tape Storage Class *configuration window may be used to further control how many tape drives/cartridges are use for migration: it will place a cap on the maximum number of VVs that can be written simultaneously in a* *storage class.*

Triggered Migration Options. There are four choices for managing migration behavior when a storage class is running out of space and the next migration isn't yet scheduled to occur:

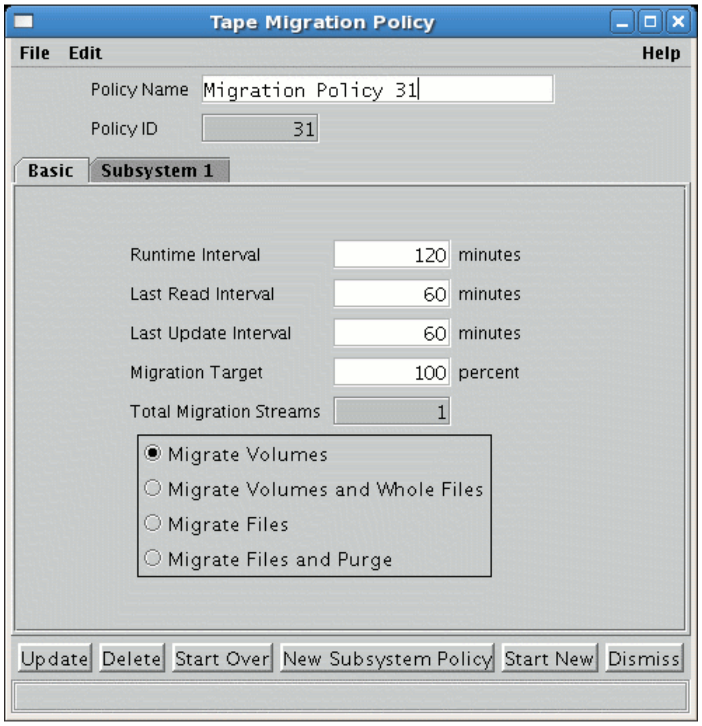
> **Migrate At Warning Threshold.** A migration run should be started immediately when the storage class warning threshold is exceeded.
>
> **Migrate At Critical Threshold.** A migration run should be started immediately when the storage class critical threshold is exceeded.
>
> **Migrate At Warning and Critical Thresholds.** A migration run should be started immediately when either the storage class warning or critical threshold is exceeded.
>
> **Migrate Only At Scheduled Intervals.** Do not trigger unscheduled migrations when storage class thresholds are exceeded.

Aggregate Files to Tape. When migrating multiple files from disk to tape, bundle them into tape aggregates (this option is ignored for disk to disk migration):

> **Min Files in Aggregate.** Minimum number of files that should be put in a single tape aggregate. If there are less than this many files to migrate, they will be migrated at a later time.
>
> **Max Files in Aggregate.** Maximum number of files that should be put in a single tape aggregate. HPSS attempts to fill up a tape aggregate with this many files. If there aren't enough candidate files, or some are ineligible, the aggregate will contain fewer files than this.
>
> **Max File Size in Aggregate.** Maximum size of file that should be put into a tape aggregate. Files larger than this will be migrated separately. Total aggregate size will not exceed this value multiplied by the Max Files in Aggregate setting. Aggregates are also limited in size to a maximum of 10% of the total estimated size of the destination tape virtual volume.

Optimize for Even File Family Migration. Select this option in order to evenly migrate files from all file families with migrateable files. Otherwise, the MPS will migrate all files within a file family before moving onward to the next file family.

6.4.2.2. Tape Migration Policy Configuration



Policy Name: Migration Policy 31
Policy ID: 31

Runtime Interval: 120 minutes
Last Read Interval: 60 minutes
Last Update Interval: 60 minutes
Migration Target: 100 percent
Total Migration Streams: 1

● Migrate Volumes
○ Migrate Volumes and Whole Files
○ Migrate Files
○ Migrate Files and Purge

This window allows an administrator to manage tape migration policies and their subsystem-specific overrides.

Subsystem-specific policies define migration rules to be applied on a subsystem basis instead of using the default (basic) migration policy. When a migration run occurs and the storage class's migration policy does not have a subsystem-specific migration policy with the same subsystem ID, then the basic migration policy will be applied; otherwise, the matching subsystem-specific migration policy will be used. Click the **Create Tape** button on the *Migration Policies* window to create a new policy. To bring up the configuration for an existing policy, select the policy from the list and click on the **Configure** button.

Field Descriptions

Policy Name. The descriptive name of the migration policy.

Policy ID. A unique ID associated with the migration policy.

Runtime Interval. The number of minutes to wait between the end of one migration run and the start of the next. This can be any positive value up to 1000000.

Last Read Interval. The number of minutes that must pass since a file was last read before it can become a candidate for migration.

Last Update Interval. The number of minutes that must pass since a file was last written before it can become a candidate for migration.

Migration Target. The desired percentage of free tape space to have available when migration is completed. The migration will be terminated when this goal is reached or when no more files meet the migration criteria. This goal may not be attainable if the total size of all files not eligible for migration is large.

Total Migration Streams. This value determines the degree of parallelism in the file migration process. This applies to policies using the **Migrate Files** and **Migrate Files and Purge** options only (see **File and Volume Options** below).

File and Volume Options. There are four options available for determining how tape migration handles files and volumes. Only one of these options may be selected at a time.

Migrate Volumes. Entire volumes are selected for migration. All segments on a selected volume will be migrated either downward in the hierarchy or laterally within the same storage level, leaving the volume ready for reclaiming.

MPS selects tape virtual volumes in EOM state with the greatest amount of vacated space. The storage segments on the selected volumes are either moved to another volume in the same storage class or are migrated downwards in the hierarchy based on the Migration Policy settings and the last read and write times for each file. When the migration step is complete, the migrated volumes' Condition should be EMPTY and the volumes should be ready to be reclaimed.

This algorithm is useful for managing space in tape storage classes without having to run repack manually. Manually reclaiming of empty volumes may still be necessary. Note that this migration algorithm moves storage segments laterally or downwards, but is not able to make multiple copies. Also, note that segments selected for migration are selected based on the volume in which they reside, not any other criteria.

A storage class configured with a migration policy using **Migrate Volumes** will not be considered as a valid alternate copy from which to retry a failed read or stage operation. No storage class to which such a storage class migrates, directly or indirectly, will be considered as a valid retry candidate, either. See the description of the **Auto Stage Retry** and **Auto Read Retry** flags in section 6.3.2, Class of Service Configuration Window on page 175.

Migrate Volumes and Whole Files. Similar to the **Migrate Volumes** option with the addition that all segments belonging to any file with a segment on the source volume will be migrated. For example, if a file consists of segments on two volumes, and one volume is selected for migration, then the segments of the file on the other volume will be migrated as well.

This algorithm tends to keep all of the segments in a file at the same level in the hierarchy and on the same volumes. Note that using this algorithm can result in large numbers of tape volumes being active at once, and hence large numbers of tape mounts.

A storage class configured with a migration policy using **Migrate Volumes and Whole Files** will not be considered as a valid alternate copy from which to retry a failed read or stage operation. No storage class to which such a storage class migrates, directly or indirectly, will be considered as a valid retry candidate, either. See the description of the **Auto Stage Retry** and **Auto Read Retry** flags in section 6.3.2, Class of Service Configuration Window on page 175.

Migrate Files. A duplicate copy of the tape files will be made at the next lower level in the hierarchy.

This enables a tape migration algorithm which is similar to disk migration. This algorithm is based on individual files rather than tape volumes, and is able to make second copies of files stored on tape. In this algorithm, individual files are selected for migration based on their last write time and the settings in the Migration Policy. The selected files are migrated downwards to the next level in the hierarchy. The order in which files are migrated is based approximately on the order in which they are written to tape. Prior to migration, however, the files are sorted by source tape volume, so their migration order may vary. In this algorithm, files at the source level are never purged. Note that the source tape storage class must be repacked and reclaimed manually.

Migrate Files and Purge. The tape files will be moved to the next lower level in the hierarchy. No duplicate copy is maintained.

This enables the same file-based tape migration algorithm as **Migrate Files**, but purges files from the source level when they are successfully migrated. Additionally, this algorithm considers both the read and write times for a candidate file against the settings in the Migration Policy. This algorithm is useful for moving inactive files to a lower hierarchy level. Note that the source tape storage class must be repacked and reclaimed manually. Also note that if migrated files are staged up in the hierarchy, HPSS does not run a purge policy to delete the cached copy at the higher level.

Related Information

Section 4.2.3.7: *Migration and Purge Policy Overrides* on page 81.

6.4.2.3. Changing a Migration Policy

To update an existing basic migration policy, select the policy from the *Migration Policies* list window and press the **Configure** button. The Disk or Tape *Migration Policy* window will appear. Make the desired changes to the basic policy and sub-system policies by selecting the appropriate tabs and placing new values in fields. When all changes have been made, press the **Update** button to record the changes.

Before changes made to a migration policy take effect, the Migration Purge Servers must be either restarted or instructed to reread the policy via the *MPS Storage Class Information* window. If the Migration Purge Servers are restarted, the changes to the migration policy are applied to all storage classes that reference the policy. If the policy is reread, the changes are only applied to the storage class and storage subsystem for which the policy is reread. It is not necessary to recycle Core Servers when the parameters in a migration policy are changed.

6.4.2.4. Deleting a Migration Policy

Before deleting a basic migration policy, make sure that it is not referenced in any storage class configuration. If a storage class configuration references a migration policy that does not exist, the Migration Purge and Core Servers will not start. Be sure to check the ***Migr Policy*** *column of the Configured Storage Classes list for any storage class that references the Migration Policy to be deleted.*

Be sure to recycle Core Servers and Migration Purge Servers after deleting a Migration Policy. Simply reinitializing them to reload the configuration is not sufficient.

To delete a migration policy, select the policy from the *Migration Policies* list window and press the **Delete** button. If a basic policy is selected, and the policy has sub-system specific policies associated with it, a prompt will appear asking if the basic policy and the related sub-system specific policies should all be deleted since they must be deleted before the basic policy can be.

A migration policy may also be deleted by pressing the **Delete** button on the *Migration Policy* configuration window for the policy. Sub-system specific policies can be individually deleted from the *Migration Policy* configuration window for the policy by selecting the specific subsystem's tab and pressing the **Delete Subsystem Policy** button.

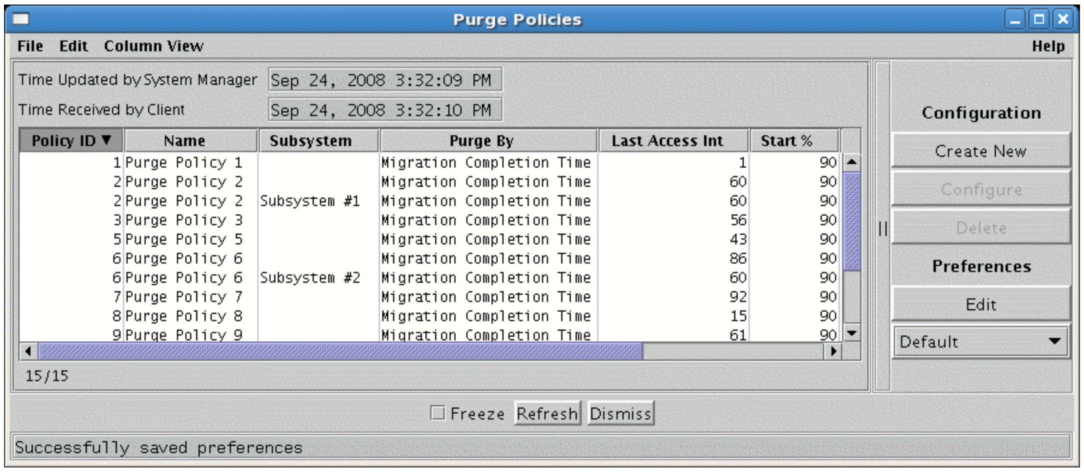
6.5. Purge Policies

A purge policy is associated with a disk storage class and defines the criteria by which data is purged from that storage class once migration has copied that data to storage classes at lower levels in the storage hierarchies.

A basic purge policy must be assigned to all disk storage classes that require purge services. This basic policy determines the purge parameters for the storage class across all storage subsystems. If individual purge policies are desired for specific storage sub-systems, storage sub-system specific purge policies may be added which override the default values in the basic policy. The storage subsystem specific purge policies share the same purge Policy ID and Policy Name with the basic policy.

Both basic and subsystem specific purge policies are created and managed using the *Purge Policies* window. The basic policy must be created before creating the subsystem specific policies.

6.5.1. Purge Policies Window



Time Updated by System Manager: Sep 24, 2008 3:32:09 PM
Time Received by Client: Sep 24, 2008 3:32:10 PM

Policy ID ▼	Name	Subsystem	Purge By	Last Access Int	Start %
1	Purge Policy 1		Migration Completion Time	1	90
2	Purge Policy 2		Migration Completion Time	60	90
2	Purge Policy 2	Subsystem #1	Migration Completion Time	60	90
3	Purge Policy 3		Migration Completion Time	56	90
5	Purge Policy 5		Migration Completion Time	43	90
6	Purge Policy 6		Migration Completion Time	86	90
6	Purge Policy 6	Subsystem #2	Migration Completion Time	60	90
7	Purge Policy 7		Migration Completion Time	92	90
8	Purge Policy 8		Migration Completion Time	15	90
9	Purge Policy 9		Migration Completion Time	61	90

15/15

Successfully saved preferences

This window displays a list of all the purge policies configured in the system.

To create a new purge policy, press the **Create New** button in the *Purge Policies* window. A new *Purge Policies* window appears. Enter the desired parameters for the policy, and then press the **Add** button. The policy will be added to the system. If sub-system specific policies are desired, press the **New Subsystem**

Policy button after the window refreshes, enter the specific purge policy parameters, and press the **Update** button. This process can be repeated for each sub-system.

When a purge policy is added to an existing storage class, the Migration Purge Servers must be restarted in order for the policy to take effect.

Field Descriptions

Purge Policy List columns. The columns provide the same information that can be found on the *Purge Policies* window in the following section.

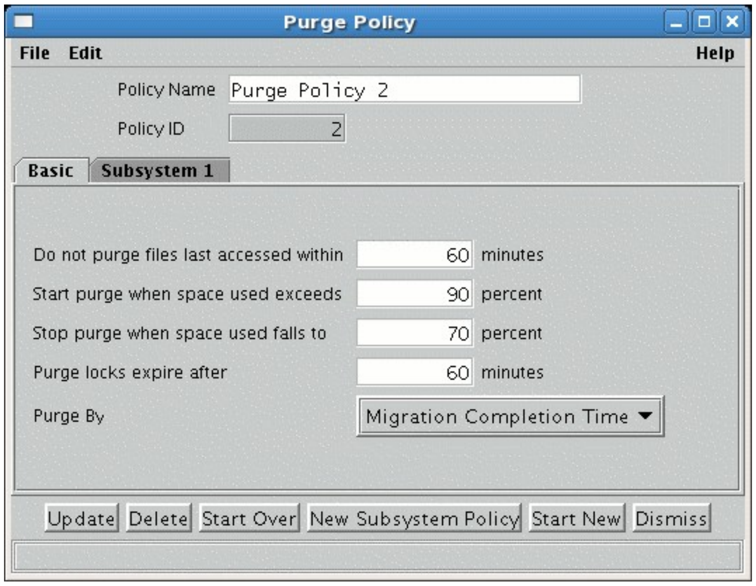
Configuration Buttons.

Create New. Open a *Purge Policies* window with default values for a new policy.

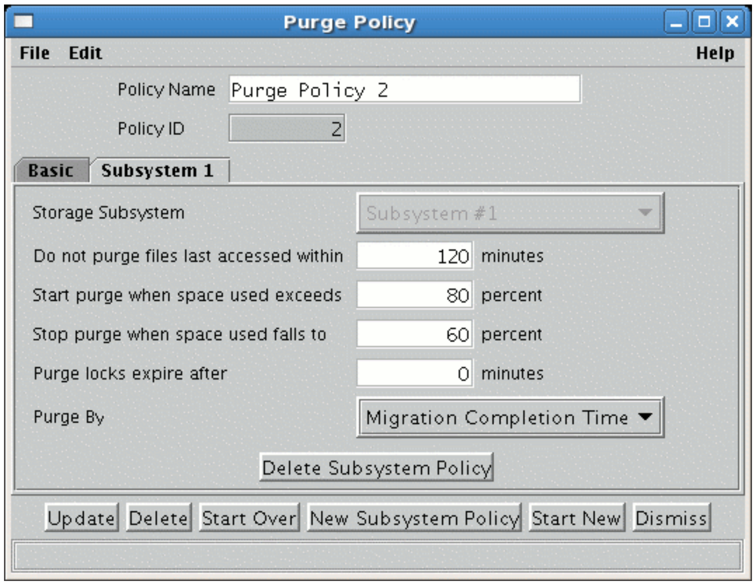
Configure. Open the selected purge policy(ies) for editing.

Delete. Delete the selected purge policy(ies).

6.5.2. Purge Policy Configuration



Policy Name: Purge Policy 2
Policy ID: 2
Do not purge files last accessed within 60 minutes
Start purge when space used exceeds 90 percent
Stop purge when space used falls to 70 percent
Purge locks expire after 60 minutes
Purge By: Migration Completion Time



Purge Policy

File Edit Help

Policy Name	Purge Policy 2
Policy ID	2

Basic | Subsystem 1

Storage Subsystem	Subsystem #1	
Do not purge files last accessed within	120	minutes
Start purge when space used exceeds	80	percent
Stop purge when space used falls to	60	percent
Purge locks expire after	0	minutes
Purge By	Migration Completion Time	

This window allows you to manage a Purge Policy. Purge policies are assigned to storage classes to tell the Migration Purge Server and Core Server how to free disk space occupied by files which have been migrated. Purge policies apply to disk storage classes only.

The window always includes the **Basic** tab and may include one or more **Subsystem** tabs. These will be referred to as "basic" and "subsystem" below.

Each purge policy consists of a single basic policy and zero or more subsystem policies. Each subsystem policy has the same policy ID and name as its basic policy. When a new purge policy is needed, the basic policy must be created first; subsystem policies can then be created as needed.

When HPSS needs to apply a purge policy to a subsystem, it first searches for a matching subsystem policy. If none is found, HPSS uses the basic policy. You can consider the basic policy to be the default for all subsystems which do not have policies of their own.

After changing any purge policy information, the changes will take effect for each Migration Purge Server (MPS) when the MPS is restarted or when it rereads the changed policy. In addition, some changes may require servers to be restarted.

When creating a subsystem policy, fields which differ from the basic policy values can be thought of as exceptions to the basic policy values. However, this does not imply that unchanged defaults maintain any links with the basic policy. Changes in the basic policy have no effect on subsystem policies.

Field Descriptions

Policy Name. The descriptive name of the Purge Policy.

Policy ID. A unique ID associated with the Purge Policy.

Do not purge files last accessed within. A file will not be a candidate for purge until it has remained unaccessed (for read or write) for the length of time specified by this field.

Start purge when space used exceeds. Purge will begin for a storage class when the amount of its space used exceeds this threshold. Used space includes any file in the storage class, whether it has been migrated or not.

Stop purge when space used falls to. Purging will stop for a storage class when the amount of its space used drops to this threshold. Note that the purge may stop before this point if it runs out of files which are available for purging.

Purge Locks expire after. Maximum number of minutes that a file may hold a purge lock. Purge locked files are not eligible for purging. A value of 0 indicates that purge locks expire immediately.

Files may be purge locked to the highest level (level zero) of a hierarchy, provided that level zero is of type disk, and data exists at level zero. Once a file is purge locked, it is no longer a purge candidate. By entering a "lock expiration duration," you will be able to monitor the number of purge locked files and the number of expired purge locked files using the Migration Purge Server (MPS) log. After a purge run, the MPS will log the total number of purge locked files and the number of expired purge locks (those files that have been locked longer than this specified number of minutes). To view the names of the files with expired purge locks, use the Purge List Utility (**plu**) with the **-lexp** parameter on a given class of service.

Purge by. The MPS uses this time attribute of a file to determine which files are eligible to be purged.

By default, files are selected for purge based on the time the purge record was created, the Migration Completion Time. Alternately, the selection of files for purging may be based on the time the file was created, the File Creation Time, or the time the file was last accessed, the Last Access Time.

If this field is changed, all Core Servers and Migration Purge Servers must be restarted in order for the change to take effect. Additionally, files will be purged in an unpredictable order until all purge records existing at the time of the change are cleared.

Storage Subsystem (subsystem policy tab only). The descriptive name of the storage subsystem to which a subsystem-specific policy applies. This field is filled in with the selected storage subsystem name at the time a subsystem specific policy is created and may not be changed afterwards.

Related Information

Section 4.2.3.7: *Migration and Purge Policy Overrides* on page 81

6.5.3. Changing a Purge Policy

To update an existing basic purge policy, select the policy from the *Purge Policies* window and click the **Configure** button. After modifying the basic policy, click on the **Update** button to save the changes. To update an existing subsystem specific purge policy, first select and display the existing basic policy and then select the tab for the desired subsystem policy. After modifying the subsystem specific policy, click on the **Update** button to save the changes.

Before changes made to a purge policy take effect, the Migration Purge Servers must be either restarted or instructed to reread the policy.[1] If the Migration Purge Servers are restarted, the changes to the purge

1 Use the Controls selection list on the MPS Storage Class Information screen or the Migration Controls selection list on the Active Storage Class list screen to reread policies.

policy are applied to all storage classes which reference the policy. If the policy is reread, the changes are only applied to the storage class and storage subsystem for which the policy is reread. Core Servers are not able to reread purge policies. If the **Purge By** field is changed on either a basic or subsystem specific purge policy, the relevant Core Servers must be restarted.

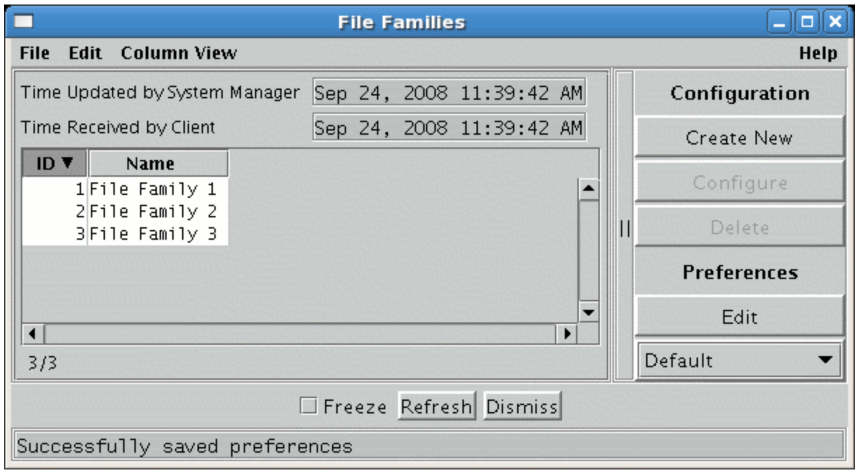
6.5.4. Deleting a Purge Policy

- *Before deleting a basic purge policy, make sure that it is not referenced in any storage class configuration. If a storage class configuration references a purge policy that does not exist, the Migration Purge and Core Servers will not start. Be sure to check the Purge Policy column of the Configured Storage Classes list for any storage class that references the Purge Policy to be deleted.*
- *Be sure to recycle Core Servers and Migration Purge Servers after deleting a Purge Policy.*

To delete a purge policy, select the policy from the *Purge Policies* window and press the **Delete** button. If a basic policy is selected, and the policy has sub-system specific policies associated with it, a prompt will appear asking if the basic policy and the related sub-system specific policies should all be deleted since the basic policy cannot be deleted until all of the sub-system specific policies are deleted.

A purge policy may also be deleted by pressing the **Delete** button on the *Purge Policy* configuration window for the policy. Sub-system specific policies can be individually deleted from the *Purge Policy* configuration window for the policy by selecting the specific subsystem's tab and pressing the **Delete Subsystem Policy** button.

6.6. File Families



Time Updated by System Manager: Sep 24, 2008 11:39:42 AM
Time Received by Client: Sep 24, 2008 11:39:42 AM

ID	Name
1	File Family 1
2	File Family 2
3	File Family 3

3/3

Successfully saved preferences

This window lists the configured file families.

Field Descriptions

The fields of the columns of this window are those of the File Family Configuration described in Section 6.6.1: *File Family Configuration*.

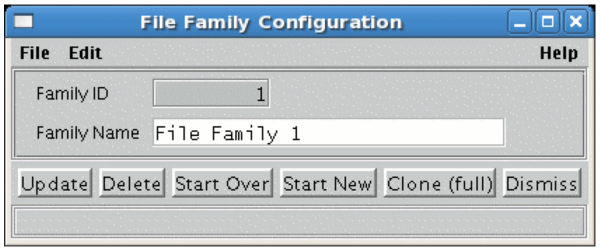
Configuration Buttons

Create New. Open a *File Family Configuration* window with default values.

Configure. Open the selected file family configuration(s) for editing.

Delete. Delete the selected file family configuration(s).

6.6.1. File Family Configuration



Family ID: 1
Family Name: File Family 1

This window allows you to manage a file family configuration.

Field Descriptions

Family ID. A positive integer which serves as a unique identifier for this file family.

Family Name. The descriptive name for this file family. The name should be unique among all file families in this HPSS installation. The default value is “File Family <ID>”.

6.6.2. Changing a File Family

The only change that can be made to a file family is to change the Family Name. This action has no side effects and takes effect when the **Update** button is pressed.

6.6.3. Deleting a File Family

SSM provides a means to delete a file family definition, but this action has no effect on files already recorded in the family. The file family attribute is permanent and cannot be removed or changed in existing files. Deleting a file family just means that files can no longer be added to the deleted file family.

Chapter 7. Device and Drive Management

Every disk and tape drive that is used by HPSS is controlled by two servers. The PVL controls mounts and dismounts (for disk devices these are logical operations only), and the Mover controls I/O. In support of these two views, the terms "PVL drive" and "Mover device" are used to refer to the configuration and information maintained about the drive by the PVL and Mover, respectively.

The configuration information is all managed through a single SSM list window. However, there are separate PVL drive and Mover device information windows, each with their own administrative state and other settings. When administrative changes are made, it is important to know whether they are to be performed against the PVL drive, the Mover device, or both.

Devices and drives are added and deleted dynamically. This means that they are added and deleted while the system is UP. Specifically the PVL must be UP; additionally it is expected that the associated Mover and associated PVR (for tape) are also UP. A new device and drive cannot be used until the PVL has successfully informed the associated Mover and PVR (for tape) about the new device and drive. Likewise, a previously deleted device and drive cannot be added back until the PVL has successfully informed the associated Mover and PVR (for tape) about the delete. Likewise, a previously added device and drive cannot be deleted until the PVL has successfully informed the associated Mover and PVR (for tape) about the add. Further details are described in this chapter.

In general, devices and drives can not be updated dynamically (i.e. while the PVL, associated Mover and associated PVR (for tape) are UP). However, there are some drive attributes for which the configuration metadata can be updated dynamically via the *PVL Drive Information* window and there are some device attributes for which the configuration metadata is not updated dynamically but the server "in memory" values can be changed dynamically (but not preserved when the Mover is restarted) via the *Mover Device Information* window. Further details are described in this chapter.

Note that all newly-created PVL drives are locked (i.e. the *Devices and Drives* window **Drive Admin State** reports "Locked"), which prevents mounts from occurring on the drive. Therefore, the PVL drive will need to be unlocked before the drive can be used. This can be done by highlighting the device/drive in the *Devices and Drives* window and pressing the "Unlock" button under the **Drive Administration** group heading.

Review the appropriate subsections of Section 5.1.1: *Physical Volume Repository (PVR) Specific Configuration* on page 109 before configuring a device/drive or modifying its state.

7.1. Configure a New Device & Drive

Be certain all tape drives are configured to use variable block sizes (Section 7.1.2: *Enable Variable Block Sizes for Tape Devices* on page 207). These steps are also covered in detail in the *HPSS Installation Guide. S*ection 3.7.7.2: *Tape Devices*.

The Device and Drive configuration entry can be created and managed using the *Devices and Drives* window (Section 7.1.1: *Devices and Drives Window* on page 202).

After a drive has been configured, its state can be changed (unlocked/locked) using the state change instructions described in Section 7.1.1: *Devices and Drives Window* on page 202.

Before adding a tape drive to HPSS, the administrator must first fully configure the tape hardware and driver software into the host platform and operating system. For disk drives, the raw disk must first be defined in the operating system. Note that using the block special file name in the device configuration may result in a large performance degradation. For SAN disk devices, the **hpss_san3p_part** utility with the -i option and device name must be run to assign a UUID to the disk device. This UUID should be used in the **Device Name** field when configuring the disk device.

Currently the number of drives which may be configured per PVR is limited to 256. The current maximum number of PVR's is 64. The maximum number of devices per mover is also 64. Given these limits, the total number of tape drives allowed in the system is 16,384.

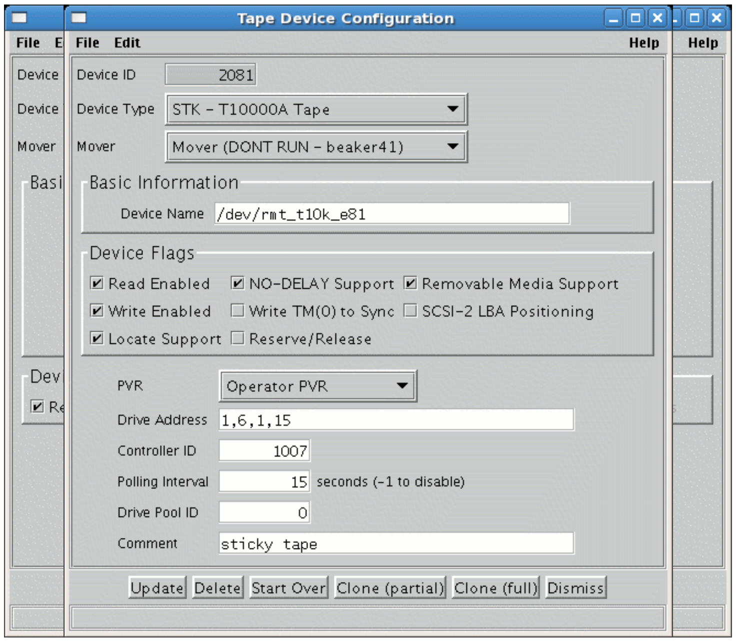
Devices and drives are added dynamically. This means that they must be added while the system is UP. Specifically the PVL must be UP; additionally it is expected that the associated Mover and associated PVR (for tape) are also UP. A new device/drive cannot be used until the PVL has successfully notified the associated Mover and PVR (for tape) about the new device/drive. If the PVL is unable to successfully notify the associated Mover and/or PVR about the new device/drive, it will retry several times; each failure will be logged as an ALARM. After the last failed attempt to notify the associated Mover and/or PVR, the PVL will abandon retrying; the administrator is advised to bounce the abandoned associated Mover (if it was UP during the failed retries) and/or to reinitialize the abandoned associated PVR (if it was UP during the failed retries) before the device/drive can be used.

Likewise, a previously deleted device/drive cannot be added back until the PVL has successfully notified the associated Mover and PVR (for tape) about the previous delete. If the administrator attempts to add a device/drive with a previously deleted Drive ID that is still pending delete notification, the administrator will get a BUSY failure and the pending notification will be aborted. An ALARM will be logged informing the administrator that the create failed since the the Mover/PVR notification pends and another ALARM will be logged informing the administrator that the previous drive delete notification will be aborted and to try later; wait a minute or so before retrying in order to give the PVL time to abort the notification. The **Drive Flag** field in the *Devices and Drives* window and *PVL Drive Information* window will report whether or not a particular drive is pending Mover/PVR notification; of course, this is only viewable while the drive exists.

At this point, the Administrative State of the drive must be unlocked before it is available. Refer to Section 7.4.1: *Unlocking a Drive* on page 220.

Disk Device Configuration

Tape Device Configuration



Device ID: 2081
Device Type: STK - T10000A Tape
Mover: Mover (DONT RUN - beaker41)

Basic Information
Device Name: /dev/rmt_t10k_e81

Device Flags
☑ Read Enabled ☑ NO-DELAY Support ☑ Removable Media Support
☑ Write Enabled ☐ Write TM(0) to Sync ☐ SCSI-2 LBA Positioning
☑ Locate Support ☐ Reserve/Release

PVR: Operator PVR
Drive Address: 1,6,1,15
Controller ID: 1007
Polling Interval: 15 seconds (-1 to disable)
Drive Pool ID: 0
Comment: sticky tape

These windows allow you to manage a tape or disk device/drive configuration.

Modifying/Updating the device/drive configuration via the *Tape Device Configuration* or *Disk Device Configuration* windows is not permitted while the PVL, the associated Mover(s), or the associated PVR(s) (for tape) are running. Updates to these configuration windows can only be attempted after these servers have been shutdown. If you are changing the Mover and/or PVR, then both the current Mover/PVR and the new Mover/PVR will need to be shutdown. Some PVL drive configuration metadata updates can occur dynamically via the *PVL Drive Information* window. Likewise, some Mover device configuration "in memory" updates (i.e. metadata not changed) can occur dynamically via the *Mover Device Information* window. Refer to Section 7.1.3: *Changing a Drive's Configuration* on page 207.

Field Descriptions

Device ID. The ID associated with this device/drive. Any positive, 32-bit integer value.

Device Type. The type of device over which data will move.

Mover. The name of the Mover that controls the device.

Advice - A single Mover can manage a maximum of 64 devices. When the mover is started and more than 64 devices are found, it will log an error and exit.

Device Name. The name by which the Mover can access the device. This name is usually the path name of a device special file such as **/dev/rmt0**. For SAN disks, this name is instead a UUID assigned to the disk device by running the hpss_san3p_part utility with the -i option and device name. The UUID for a SAN disk device can be determined by running the hpss_san3p_part utility with the -s option.

Advice - For locally attached disk devices, the pathname should refer to the raw/character special file (e.g., /dev/rhpss_disk1).

For AIX systems, SCSI attached tape drives are typically referred to by pathnames of the form /dev/rmtX, where X begins at zero and is incremented for each tape drive detected.

For IRIX systems, SCSI attached tape drives are typically referred to by pathnames of the form /dev/rmt/ tpsXdYns, where X is the SCSI controller number, and Y is the SCSI ID of the drive. Note that for Ampex DST drives, the tpsXdYnrns name should be used (indicating that the driver should not attempt to rewind the drive upon close). For other drives on IRIX, the tpsXdYnsvc name should be used (indicating that the driver allows compression and variable block sizes).

For Solaris systems, SCSI attached tape drives are typically referred to by pathnames of the form /dev/rmt/Xc, where X begins at zero and is incremented for each tape drive detected (the 'c' indicates that compression is enabled). In particular note that the device that contains a 'b' in the name should NOT be used, as this will change the behavior of the drive and cause the HPSS Mover to fail.

For Linux systems, this is the name that will be used to provide access to the SCSI raw device. The pathname will be in the form /dev/raw/rawX, where X specifies the raw device number. You can run the "raw -q -a" command to determine the correct raw device mappings. SCSI tape devices are referred to by pathnames of the form /dev/stX, where X begins at zero and is incremented for each LUN detected.

Media Block Size (disk only). The block size for the device. This value should be a multiple of the underlying disk block size; otherwise an error will occur at first I/O.

Bytes on Device (disk only**).** The size of the device in bytes.

Advice - The storage class to which this drive will be assigned must have a PV Size less than or equal to this value.

If the Starting Offset is non-zero, then the Bytes on Device value cannot be greater than the actual size of the underlying device less the Starting Offset value.

If this value is modified after the disk has been imported into HPSS, it must be emptied, exported and re-imported.

Starting Offset (disk only**).** The offset in bytes from the beginning of the disk logical volume at which the Mover will begin using the volume. The space preceding the offset will not be used by HPSS. This value should be a multiple of the **Media Block Size**.

Advice - In some cases, the operating system may maintain control information at the beginning of a disk volume which should not be overwritten by HPSS. As a specific example, the AIX Logical Volume Manager (LVM) currently writes a control block at the start of a logical volume that is required during some operations on mirrored logical volumes (such as splitting the mirror in the case of a failing physical disk). The Starting Offset field can be used to force the Mover to skip over that control block before writing the HPSS volume label or any user data (consult AIX documentation to determine the size of the LVM control block).

SAN ID (disk only**).** The ID for the SAN group. A group is a set of devices that are logically related, with the primary characteristic being that all devices in group are accessible by all clients. Each group is identified with this globally unique group id. It is assigned to all devices within a SAN group.

Device Flags. The following fields are the device flags used by the Mover.

- **Read Enabled**. An indication of whether the device is available for reading.
- **Write Enabled**. An indication of whether the device is available for writing.
- **Locate Support** (tape only). An indication of whether the device supports a high speed (absolute) positioning operation.

Advice - This option is supported for 3590, 3590E, 3580, 3592, 9840, 9940, DST-312, DST-314, T10000 and GY-8240 devices.

- **NO-DELAY Support** (tape only)**.** An indication of whether the device supports opening the device with no delay flag set, while allowing tape I/O operation after the open.

  Advice - On some tape devices, this will allow for a quicker polling operation when no tape is presently loaded in the device. This field is meaningful for tape devices only.

- **Write TM(0) to Sync** (tape only)**.** An indication of whether the device supports issuing a write tape mark request with a zero count to flush data written previously to the tape media.

  Advice - On some devices, this may provide a higher performance method to ensure that data has been safely written to the media. This option is not used for the IBM 3590 and 3590E devices, provided the HPSS supported device driver is used in accessing the device. Note that for Ampex DST-312 this field should be set to "ON".

- **Removable Media Support** (tape only)**.** An indication of whether the device supports removable media.

- **SCSI-2 LBA Positioning** (tape only). If ON, the tape device supports SCSI-2 Logical Block Addresses.

  Advice - If SCSI-2 LBAs and the SCSI LOCATE command (Locate Support) are supported by the device, HPSS will calculate tape read addresses based on known LBAs and relative addresses. LBA positioning provides for faster access of data residing on tape. The benefit will be realized for read requests with many source descriptors specifying locations spread sparsely down the tape. This is supported only by the IBM SCSI tape device driver.

- **SAN3P-transfer Enabled** (disk only). If ON, SAN3P data transfers to this device will be supported.

  Warning *– There is a security vulnerability associated with the use of SAN3P. If a user is root on a machine which has access to the SAN (e.g. a client machine) then that user has the potential to access or destroy fiber-channel connected disk storage. Two areas of concern: 1) verification that only authorized users (usually limited to only 'root' or 'hpss') are granted read and write access to these resource; 2) HPSS administrators should be aware that machines, possibly owned or managed by other groups, which are added to the SAN to facilitate the use of SAN3P transfers will have access to all data on disk and tape resources. If those systems are compromised, or there are individuals authorized for systems privileges on those particular machines, but not necessarily authorized for HPSS administrative access, there is the potential for access and/or damage to HPSS data. These are inherent limitations of SAN implementations that have not yet been addressed by the industry and cannot be remedied by HPSS.*

- **Multiple Mover Tasks** (disk only). If ON, the Mover will allow multiple Mover tasks to access the disk device.

- **Reserve/Release** (tape only). An indication of whether a SCSI reservation is taken on the device when it's opened.

  Advice - This is useful on fiber attached tape devices to ensure that HPSS has sole control on the *device. Without the reservation, it is possible for other hosts to interleave SCSI commands to the drive with those issued by HPSS. This effect could potentially lead to corruption of data.*

Table 2. Recommended Settings for Tape Devices

Device Driver	Mover Executable	NO-DELAY Support	Locate Support	Write TM(0) to Sync
IBM SCSI Tape	hpss_mvr_ssd	ON	ON	OFF
IRIX Native	hpss_mvr_tcp	ON	ON	OFF
Ampex	hpss_mvr_dd2	OFF	ON	ON
AIX Native	hpss_mvr_tcp	ON	OFF	OFF
Linux Native	hpss_mvr_tcp	ON	ON	OFF

PVR (tape only)**.** The name of the PVR that handles the removable media operations for the drive.

Drive Address. For tape drives, this is the name/address by which the PVR can access the drive. This field isn't used by the PVL or PVR for disk drives; the administrator may use this as a comment. Many sites duplicate the **Device Name** which makes it valuable as a query of drive information since it will return the device information for the disk case (e.g. `lshpss -drv`).

Advice - For StorageTek robots: Drive Address configuration entries correspond to the ACS,Unit,Panel,Drive Number used by ACSLS to identify drives. For example, the first drive in a typical configuration has Drive Address 0,0,10,0.

For IBM 3584 LTO robots: The Drive Address configuration entries correspond to the SCSI address (location) of the drive in the library. Determine the drive location by running an inventory on the library either through the tapeutil utility supplied with the Atape driver or using the library console. Typically the first drive is located at location 257 for a single frame library.

For IBM 3494 robots: The Drive Address configuration entries correspond to the hexadecimal Library device number of the drive. Determine the Library device number by running the command "/opt/hpss/bin/GetESANumbers /dev/rmtX" for each tape drive in the robot.

For SCSI PVRs: The Drive Address configuration entries correspond to the SCSI address (location) of the drive in the library. Determine the drive location by running an inventory on the library either through the device_scan utility or using the library console. Typically the first drive is located at location 256 for a single frame library.

For operator mounted drives: For manually mounted drives, the drive address string is displayed to operators when an explicit drive is selected for mounting. The drive address should therefore be a string that easily communicates to an operator the drive in question (i.e., a name matching the label on the exterior of the drive).

For AML robots: Leave the Drive Address field blank.

The AML PVR is supported by special bid only.

Controller ID. An indication of the adapter/bus that is in the data path to the device.

Advice - This field is used to attempt to mount individual volumes of a set of striped media on individual controllers, when possible, to prevent contention on the data path. It is recommended that the drives that are on the same adapter/bus have the same Controller ID. For drives in 3494 robots, the Controller ID *must be a valid ID. The valid IDs can be found in the Affinity list for any cartridge in the robot. Use the command "mtlib -l <device name> -qV -V<volume name>" to obtain the Affinity list for a cartridge.*

Polling Interval (tape only). The number of seconds to wait between polling requests performed by the PVL to determine if any media is present in the drive. Use -1 to disable polling. Values of 0 to 14 are not valid.

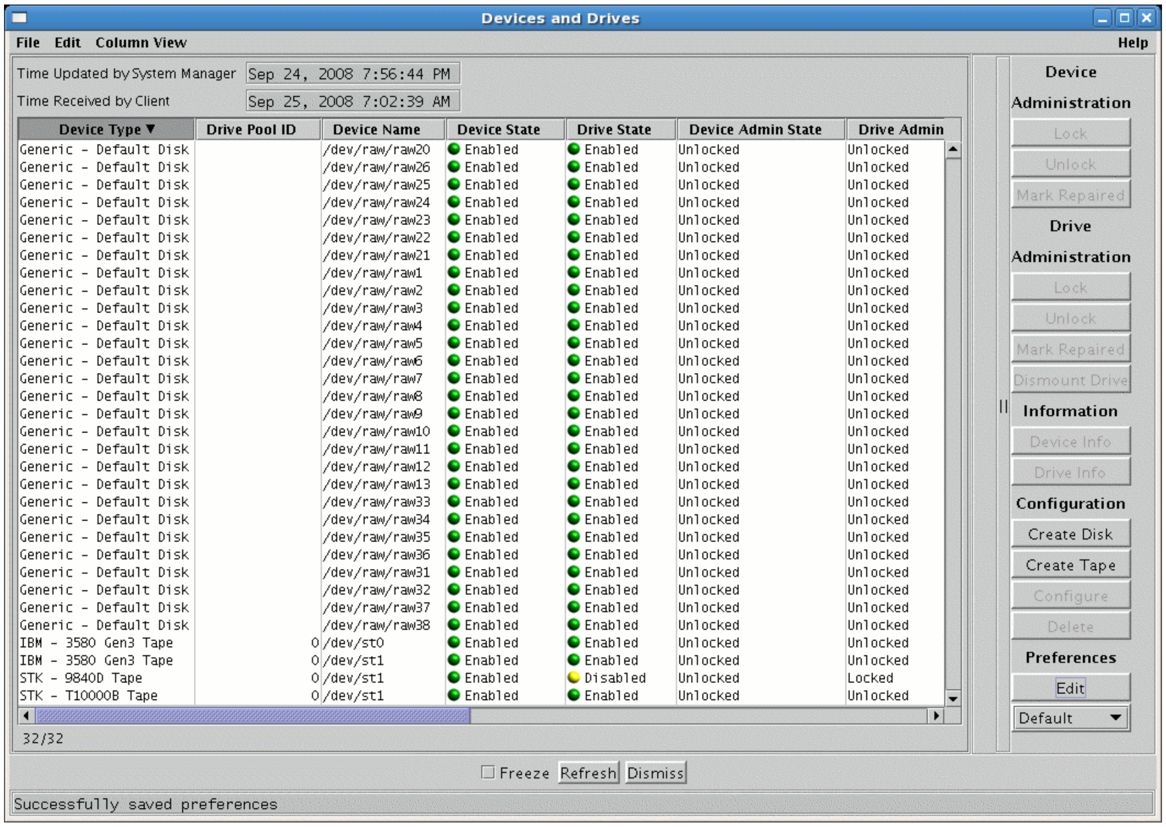
Advice - The value for tape drives located within robotic libraries should be set higher than for manually mounted drives. It is recommended that this field be set to 15 seconds for operator-mounted tape drives and 900 (15 minutes) for library-managed tape drives.

Mounted Volume (disk only). The 8-character name of the volume mounted on the disk drive.

Drive Pool ID (tape only). If non-zero positive integer value, the **Drive Pool ID** will restrict this drive to only be scheduled for tape read requests specifying this value. The default **Drive Pool ID** is 0 to indicate that the drive is not restricted to a particular drive pool. Negative values are not allowed. See Section 7.3: *Drive Pools* on page 218.

Comment. This field provides a 128 character buffer in the PVL drive metadata which gives the administrator the opportunity to associate miscellaneous text with a device/drive. For example, a site may want to place a comment in this field that the drive is out of service or being used by another system, etc.

7.1.1. Devices and Drives Window



Time Updated by System Manager: Sep 24, 2008 7:56:44 PM

Time Received by Client: Sep 25, 2008 7:02:39 AM

Device Type ▼	Drive Pool ID	Device Name	Device State	Drive State	Device Admin State	Drive Admin
Generic - Default Disk		/dev/raw/raw20	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw26	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw25	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw24	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw23	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw22	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw21	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw1	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw2	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw3	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw4	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw5	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw6	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw7	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw8	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw9	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw10	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw11	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw12	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw13	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw33	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw34	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw35	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw36	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw31	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw32	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw37	● Enabled	● Enabled	Unlocked	Unlocked
Generic - Default Disk		/dev/raw/raw38	● Enabled	● Enabled	Unlocked	Unlocked
IBM - 3580 Gen3 Tape	0	/dev/st0	● Enabled	● Enabled	Unlocked	Unlocked
IBM - 3580 Gen3 Tape	0	/dev/st1	● Enabled	● Enabled	Unlocked	Unlocked
STK - 9840D Tape	0	/dev/st1	● Enabled	● Disabled	Unlocked	Locked
STK - T10000B Tape	0	/dev/st1	● Enabled	● Enabled	Unlocked	Unlocked

32/32

Successfully saved preferences

This window allows you to view the list of configured Mover devices and PVL drives. It also provides a number of function buttons, which allow certain operations to be performed on devices or drives.

The *Device and Drive List Preferences* window may be used to select the device/drives which will be displayed. Select the columns to be displayed in the list from the Column View menu.

Most of the function buttons to the right of this list require that one or more device/drive entries be selected from the list. Note that some of the fields in the table may not be visible, depending on the Column View menu settings.

Field Descriptions

ID. The unique device/drive ID number defined when the device/drive is configured.

Device Type. The type of the device.

Device Name. The name used by the Mover to access the device, usually the pathname of a UNIX device special file (such as "/dev/rmt0").

Device State. The current operational state of the device, as reported by its controlling Mover. The possible states are:

- Enabled - The device is working normally.
- Suspect - The Mover has detected errors on the device, but it is still functioning.

- Disabled - The device is locked, which makes it unavailable for use by HPSS.
- Unknown - The state of the device is not known to SSM; this is usually caused by the controlling Mover being down or disconnected from SSM.

Device Admin State. The current administrative state of the device, as reported by its controlling Mover. The possible states are:

- Locked - The device is locked and unavailable for use by HPSS.
- Unlocked - The device is unlocked and available for use by HPSS.
- Unknown - The state of the device is not known to SSM; this is usually caused by the controlling Mover being down or disconnected from SSM

Mover. The descriptive name of the Mover that controls the device.

Mover Host. The host name of the Mover node (the node where the Mover runs).

Drive Address. The name used by the PVL/PVR to access the drive.

Drive Flag. This field describes possible actions that the PVL needs to perform for the drive. Possible flags are:

- Clear - Nothing needs to be done.
- Modified - Set by SSM when the drive has been modified. It was added for a future feature; currently unsupported.
- PVR/Mover Notify Pending - The PVL needs to notify the associated PVR and Mover that the drive has been created or deleted.
- PVR Notify Pending - The PVL needs to notify the associated PVR that the drive has been created or deleted.
- Mover Notify Pending - The PVL needs to notify the associated Mover that the drive has been created or deleted.
- Abort PVR/Mover Notify – The PVL is aborting a pending notification.

Drive State. The current operational state of the drive, as reported by the PVL. The possible states are:

- Enabled - The drive is working normally.
- Disabled **-** The drive is not available for use by HPSS. This can occur if the drive is locked by an SSM user, or if an error occurred when mounting or dismounting a cartridge on the drive.
- Broken **-** The PVL itself has detected a fatal error, and is shutting itself down.
- Unknown **-** The state of the device is not known to SSM; this is usually caused by the PVL being down or disconnected from SSM.

Drive Admin State. The current administrative state of the drive, as reported by the PVL. The possible states are:

- Locked - The drive is locked and unavailable for use by HPSS.
- Unlocked - The drive is unlocked and available for use by HPSS.

- Unknown - The state of the drive is not known to SSM; this is usually caused by the PVL being down or disconnected from SSM

Comment. This field provides a 128 character buffer in the PVL drive metadata which gives the administrator the opportunity to associate miscellaneous text with a device/drive. For example, a site may want to place a comment in this field that the drive is out of service or being used by another system, etc.

PVR. The descriptive name of the PVR that handles removable media operations for the drive. This will be blank if the device/drive does not support removable media.

Mounted Volume. The name of the volume that is currently mounted as reported by the PVL.

Bytes Read. Number of bytes read from the device/drive as reported by the Mover.

Bytes Written. Number of bytes written to the device/drive as reported by the Mover..

State. The overall operational state of the device/drive. If this is not Enabled, please check the individual **Device State** and **Drive State** fields to help determine why.

Drive Pool ID. If non-zero positive integer value, the **Drive Pool ID** will restrict this drive to only be scheduled for tape read requests specifying this value. The default **Drive Pool ID** is 0 to indicate that the drive is not restricted to a particular drive pool. Negative values are not allowed. See Section 7.3: *Drive Pools* on page 218. This field is not applicable for disks and thus will display nothing for disks.

Device Administration Buttons

This group of buttons affects selected Mover devices. All the buttons are disabled unless one or more devices are selected (see figure above).

Lock. Locking the selected devices serves no functional purpose. When the device is locked, the information is not transferred to the PVL and thus the PVL will continue to schedule the associated PVL drive (see **Lock** button in the **Drive Administration Buttons** below). It is also not transferred to the Core Server (CS) where the CS will continue to read/write/create/mount the Volume (see **VV Condition** in the *Core Server Volume Disk/Tape* window). This leads to many red Alarms. After pressing the Device Administration **Lock** button, you are prompted to confirm your request before it is actually executed. The window's message line reports when locking begins. When the request is complete, another message tells how many requests were successful.

Unlock. Unlock the selected devices, making them available to the Mover. You are prompted to confirm your request before it is actually executed. The window's message line reports when unlocking begins. When the request is complete, another message tells how many requests were successful.

Mark Repaired. Marks the selected devices as “repaired”. Sometimes the device states (such as operational state) will continue to indicate an error condition after the cause of the error has been fixed. If so, you can mark a device repaired to instruct the Mover to clear its error states. Note that this does nothing, either in hardware or software, to actually repair an error condition. Also, if you mark a device repaired when it still has a problem, the error states will be cleared but may quickly return to their previous values.

You are prompted to confirm your request before it is actually executed. The window's status bar displays a message when marking begins, and when the request is complete, another message tells how many requests were successful.

Drive Administration Buttons

This group of buttons affects selected drives. All the buttons are disabled unless one or more drives are selected (see figure above).

Lock. Lock the selected drives, making them unavailable to HPSS. When a drive is locked, the PVL will no longer schedule the PVL drive. When locking a tape drive due to a cartridge problem (e.g. stuck tape), it is beneficial to also cancel the PVL Job associated with the cartridge. Additionally, locking the drive is not transferred to the Core Server (CS), therefore the CS will continue to read/write/create/mount the Volume (see **VV Condition** in the *Core Server Volume Disk/Tape* window) unless you change the **VV Condition** to DOWN (or EOM). After pressing the Drive Administration **Lock** button, you are prompted to confirm your request before it is actually executed. The window's message line reports when locking begins. When the request is complete, another message tells how many requests were successful.

Unlock. Unlock the selected drives, making them available to HPSS. You are prompted to confirm your request before it is actually executed. The window's message line reports when unlocking begins. When the request is complete, another message tells how many requests were successful.

Mark Repaired. Notify the PVL and Mover to clear the error states for the selected drives. Sometimes the drive states (such as operational state) will continue to indicate an error condition after the cause of the error has been fixed. If so, you can mark a drive repaired to instruct the Mover or PVL to clear its error states. Note that this does nothing, either in hardware or software, to actually repair an error condition. Also, if you mark a drive repaired when it still has a problem, the error states will be cleared but may quickly return to their previous values.

You are prompted to confirm your request before it is actually executed. The window's status bar displays a message when marking begins, and when the request is complete, another message tells how many requests were successful.

Dismount. Dismount the selected drives. You are prompted to confirm your request before it is actually executed. The window's status bar displays a message when dismounting begins. When the request is complete, another status message tells how many requests were successful.

Information Buttons

This group of buttons retrieves the Device or Drive managed object for one or more selected list items. All the buttons are disabled unless one or more devices/drives are selected.

Device Info. Open the *Mover Device Information* window for the selected item. If the selected item’s Mover cannot be contacted, this button will be desensitized. While this window is open, it will continue to update reflecting the latest device data from the Mover.

Drive Info. Open the *PVL Drive Information* window for the selected item. If the PVL cannot be contacted, this button will be desensitized. While this window is open, it will continue to update reflecting the latest drive data from the PVL.

Configuration Buttons

This group of buttons allows you to do device/drive configuration.

Create Disk. This button is always active. Clicking on it opens the *Disk Device Configuration* window in Add mode, allowing you to create a new disk device and drive.

Create Tape. This button is always active. Clicking on it opens the *Tape Device Configuration* window in Add mode, allowing you to create a new tape device and drive.

Configure. This button is active if one or more devices/drives are selected. Clicking the button opens the *Disk/Tape Device Configuration* window for the selected devices and drives, allowing you to view, delete, clone, or update the configuration(s).

Delete. This button is active if one or more device/drives is selected from the list. A dialog will be displayed for each device/drive selected asking the user to confirm that the device/drive configuration is to be deleted.

Preferences

See Section 7.1.1: *Devices and Drives Window* on page 202 for information on changing preferences for the *Devices and Drives* window.

7.1.2. Enable Variable Block Sizes for Tape Devices

All tape devices that will be used for HPSS data must be set to handle variable block sizes (to allow for the ANSI standard 80-byte volume label and file section headers). This requirement is described in detail in the *HPSS Installation Guide,* Section 3.7.7.2: *Tape Devices.* Please refer to that section to be certain your tape devices are configured correctly.

7.1.3. Changing a Drive's Configuration

In general, devices and drives can not be **updated** dynamically (i.e. while the PVL, associated Mover(s) and associated PVR(s) (for tape) are UP). However, there are some drive attributes for which the configuration metadata can be updated dynamically via the *PVL Drive Information* window and there are some device attributes for which the configuration metadata is not updated dynamically but the server "in memory" values can be changed dynamically (but not preserved when the Mover is restarted) via the *Mover Device Information* window.

The device and drive configuration entry can be updated using the *Disk Device Configuration* or *Tape Device Configuration* window (Section 7.1.1: *Devices and Drives Window* on page 202). The *Disk Device Configuration* and *Tape Device Configuration* windows can **not** be updated dynamically. These windows are opened by pressing the Configure button on the *Devices and Drives* window. Changing any settable field on the *Disk/Tape Device Configuration* windows requires that several servers be shutdown:

- the PVL
- the Mover that is currently managing the device
- the new Mover that will manage the device (if changing)
- the PVR that is currently managing cartridges for this drive (if tape)
- the PVR that will manage cartridges for this drive (if tape and if changing)

If any of these servers are running, follow the instructions in Section *5.1.1: Shutting Down an HPSS Server* on page 151 to shutdown the servers. Verify that these servers are no longer running before attempting to reconfigure the drive. SSM will display a warning listing the servers requiring shutdown to help the administrator verify that the proper servers have been shutdown.

Some PVL drive configuration attributes can be updated dynamically using the *PVL Drive Information* window (Section 7.2.2: *PVL Drive Information Window* on page 214). The settable fields in this window are updated dynamically (i.e. saved to metadata and used by the PVL upon successful Update).

It is also possible to make temporary updates to some Mover device attributes parameters by changing the settable fields in the *Mover Device Information* window (Section 7.2.1: *Mover Device Information Window* on page 209) instead of the configuration window. These changes stay in effect until the Mover is recycled.

To help the administrator update device/drive configuration data that is not settable dynamically (e.g. the **Mover** field), a useful workaround is to delete and then add the device/drive back. (Please review the procedure/rules for deleting and adding devices/drives before attempting this procedure.) For example:

- Open the *Disk/Tape Device Configuration* window for the device/drive you want to change. [Do this by selecting the device/drive that you want to update in the *Devices and Drives* window and then press the **Configure** button.]
- Press the **Clone (full)** button on the *Disk/Tape Device Configuration* window that you just opened [leave this window open]
- Verify the drive can be deleted. Refer to Section 7.1.4: *Deleting a Drive's Configuration* on page 208.
- Use the *Devices and Drives* window to select the same device/drive and press the **Lock** button
- Use the *Devices and Drives* window to select the same device/drive and press the **Delete** button
- Go back to the 'cloned' window and modify the fields requiring changes (e.g. Mover field)
- Press the **Add** button

7.1.4. Deleting a Drive's Configuration

Devices and drives are deleted dynamically. This means that they must be deleted while the system is UP. Specifically the PVL must be UP; additionally it is expected that the associated Mover and associated PVR (for tape) are also UP since the PVL needs to notify these servers about the deletion. If the PVL is unable to successfully notify the associated Mover and/or PVR about the deleted device/drive, it will retry several times; each failure will be logged as an ALARM. After the last failed attempt to notify the associated Mover and/or PVR, the PVL will abandon retrying; the administrator is advised to bounce the abandoned associated Mover (if it was UP during the failed retries) and/or to reinitialize the abandoned associated PVR (if it was UP during the failed retries) to ensure that the device/drive is completely deleted and thus can no longer be used.

Likewise, a previously added device/drive cannot be deleted until the PVL has successfully notified the associated Mover and PVR (for tape) about the previous add. If the administrator attempts to delete a device/drive with a previously added Drive ID that is still pending add notification, the administrator will get a BUSY failure and the pending notification will be aborted. An ALARM will be logged informing the administrator that the delete failed since the Mover/PVR add/create notification is pending. Another ALARM will be logged informing the administrator that the previous drive create notification will be aborted and to try later; wait a minute or so before retrying in order to give the PVL time to abort the notification. The **Drive Flag** field in the *Devices and Drives* window and *PVL Drive Information* window will report whether or not a particular drive is pending Mover/PVR notification; of course, this is only viewable while the drive exists.

There are a number of situations in which the PVL won't allow the device/drive to be deleted:

- If the device/drive is still attempting to notify the Mover/PVR about being added
- If the device/drive is in the process of aborting Mover/PVR notification
- If the drive is in use by the PVL
- If the drive is not locked
- For disk: If storage resources haven't been deleted from the disk device/drive
- For disk: If the physical volume hasn't been exported from the disk device/drive

If it is only necessary to take a drive out of service for a finite period of time, the administrator should follow the instructions in Section 7.1.1: *Devices and Drives Window* on page 202 to lock the drive and prevent its use rather than permanently removing the drive as described in this section.

The Device and Drive configuration entry can be deleted using the *Devices and Drives* list window (Section 7.1.1: *Devices and Drives Window* on page 202). Select the correct device/drive configuration and click on the Delete button to delete the configuration. Once the drive configuration is deleted, HPSS will no longer be able to use the drive's resources.

7.2. Monitoring Devices and Drives

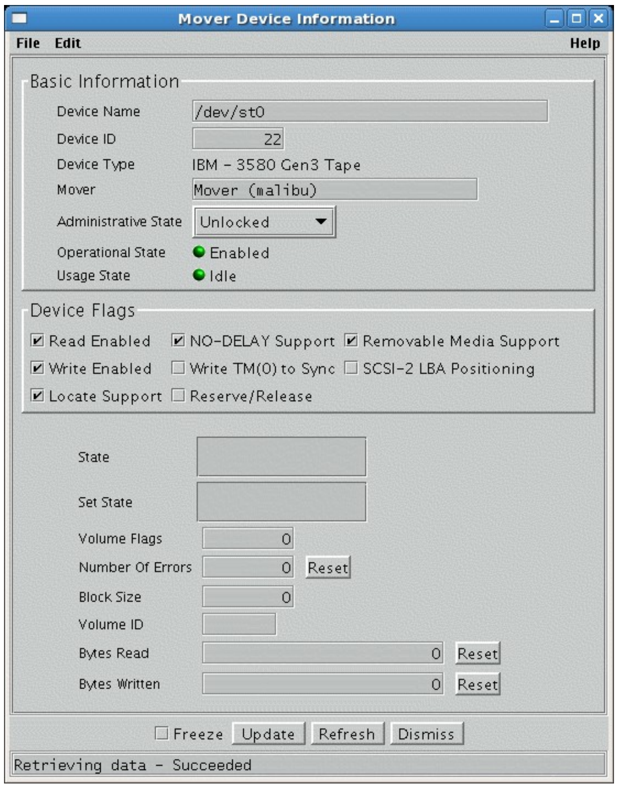
After the devices/drives are configured and are made available for service, the device/drive status may be monitored via several SSM windows.

When a device/drive experiences a problem, the PVL and/or the Mover issues one or more alarms to inform SSM users of the problem and to notify SSM of the changes in the device/drive state. SSM reports the state change on the Device/Drives Status field on the *HPSS Health and Status* window. SSM also reports the new state on the *Devices and Drives* list window, the *Mover Device Information* window, and the *PVL Drive Information* window if they are being displayed at the time.

If an HPSS device or drive needs to be locked or unlocked or if any device/drive configuration needs to be done, the *Devices and Drives* list window is the first stop (for detailed information, see Section 7.1.1: *Devices and Drives Window* on page 202). It facilitates drive lock/unlock and also lets administrators access the following windows:

- *Disk/Tape Device Configuration* - used to configure/modify/delete device and drive configurations. See Section 7.1: *Configure a New Device & Drive* on page 196 for more information.
- *Mover Device Information* - used to view or update the state of the Mover device. See Section 7.2.1: *Mover Device Information Window* for more information.
- *PVL Drive Information* - used to view or update the state or configuration of the PVL drive. See Section 7.2.2: *PVL Drive Information Window* on page 214 for more information.

7.2.1. Mover Device Information Window



Basic Information

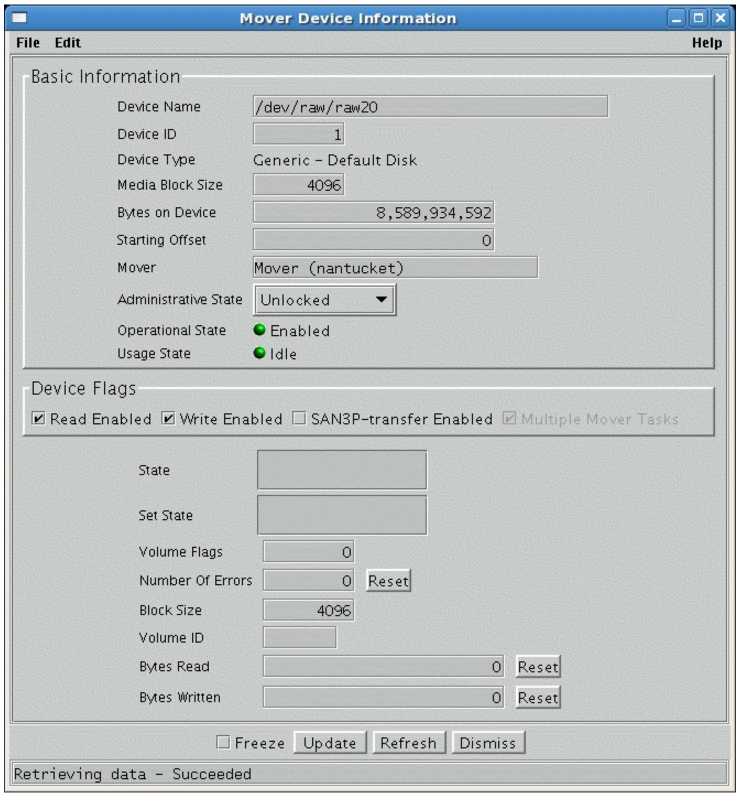
Device Name	/dev/st0
Device ID	22
Device Type	IBM - 3580 Gen3 Tape
Mover	Mover (malibu)
Administrative State	Unlocked
Operational State	● Enabled
Usage State	● Idle

Device Flags

☑ Read Enabled ☑ NO-DELAY Support ☑ Removable Media Support
☑ Write Enabled ☐ Write TM(0) to Sync ☐ SCSI-2 LBA Positioning
☑ Locate Support ☐ Reserve/Release

State	
Set State	
Volume Flags	0
Number Of Errors	0
Block Size	0
Volume ID	
Bytes Read	0
Bytes Written	0

Retrieving data - Succeeded



Basic Information

Device Name	/dev/raw/raw20
Device ID	1
Device Type	Generic - Default Disk
Media Block Size	4096
Bytes on Device	8,589,934,592
Starting Offset	0
Mover	Mover (nantucket)
Administrative State	Unlocked
Operational State	● Enabled
Usage State	● Idle

Device Flags

☑ Read Enabled ☑ Write Enabled ☐ SAN3P-transfer Enabled ☑ Multiple Mover Tasks

State	
Set State	
Volume Flags	0
Number Of Errors	0
Block Size	4096
Volume ID	
Bytes Read	0
Bytes Written	0

Retrieving data - Succeeded

The *Mover Device Information* window reports the current statistics for the device, such as the workload history of the device since the startup of the controlling Mover. The *Mover Device Information* window can also be used to lock and unlock a mover device (note: locking the Mover device generally is not helpful; see S7.1.1ct*Devices and Drives Window*on : o202 page). Additionally, it can be used to control the I/O aspects of the device.

Changes to the Administrative State cause a confirmation window to pop up. If the user confirms the request, modifications made to all fields on the window will be sent to the Mover and become effective immediately.

Field Descriptions

Device Name. The name by which the Mover accesses the device; usually the pathname of a UNIX device special file (such as "/dev/rmt0").

Device ID. The ID number unique to this device. The same ID number also refers to the corresponding PVL drive.

Device Type. The HPSS media type which describes the physical device.

Media Block Size (disk only). The block size for the device.

Bytes on Device (disk only). The size of the disk device in bytes.

Starting Offset (disk only). The offset in bytes from the beginning of the disk logical volume at which the Mover will begin using the volume. The space preceding the offset will not be used by HPSS.

Mover. The descriptive name of the Mover which controls the device.

Administrative State. This field allows you to modify the state of the device. Click on the option menu button to pop up a list of valid states. These are:

- Locked - Makes the device unavailable for HPSS requests.
- Unlocked - Makes the device available for HPSS requests.
- Mark Repaired - Tells the Mover to clear any error status for the device. This can be useful if you think a problem has been fixed, but the Mover is unaware of it. This does not do anything, in hardware or software, to fix a problem; it only clears error indicators.

Operational State. Indicates the device's ability to handle HPSS requests. Possible values for this field are:

- Enabled - The device is available for HPSS requests.
- Disabled - The device is unavailable for HPSS requests, possibly caused by setting the "Locked" Administrative State.

Usage State. Indicates the state of the Mover's control over the device. Possible values for this field are:

- Active - A Mover task currently has control of the device.
- Idle - No Mover task is controlling the device.

Device Flags. This group of buttons defines various device characteristics. When a button is ON, it means that the corresponding flag is set.

- **Read Enabled**. An indication of whether the device is available for reading.
- **Write Enabled**. An indication of whether the device is available for writing.
- **Locate Support** (tape only). An indication of whether the device supports a high speed (absolute) positioning operation.

  Advice - This option is supported for IBM 3590, 3590E, 3580, 3592, StorageTek 9840, 9940, DST-312, DST-314, T10000 and GY-8240 devices.

- **NO-DELAY Support** (tape only)**.** An indication of whether the device supports opening the device with no delay flag set, while allowing tape I/O operation after the open.

  Advice - On some tape devices, this will allow for a quicker polling operation when no tape is presently loaded in the device. This field is meaningful for tape devices only.

- **Write TM(0) to Sync** (tape only)**.** An indication of whether the device supports issuing a write tape mark request with a zero count to flush data written previously to the tape media.

  Advice - On some devices, this may provide a higher performance method to ensure that data has been safely written to the media. This option is not used for the IBM 3590, and 3590E devices, provided the HPSS supported device driver is used in accessing the device. Note that for Ampex DST-312 this field should be set to "ON".

- **Removable Media Support** (tape only)**.** An indication of whether the device supports removable media.
- **SCSI-2 LBA Positioning** (tape only). If ON, the tape device supports SCSI-2 Logical Block Addresses.

  Advice - If SCSI-2 LBAs and the SCSI LOCATE command (Locate Support) are supported by the device, HPSS will calculate tape read addresses based on known LBAs and relative addresses. LBA positioning provides for faster access of data residing on tape. The benefit will be realized for read requests with many source descriptors specifying locations spread sparsely down the tape. This is only supported by the IBM SCSI tape device driver.

- **SAN3P-transfer Enabled** (disk only). If ON, SAN3P data transfers to this device will be supported.

  Warning *– There is a security vulnerability associated with the use of SAN3P. If a user is root on a machine which has access to the SAN (e.g. a client machine) then that user has the potential to access or destroy fiber-channel connected disk storage. Two areas of concern: 1) verification that only authorized users (usually limited to only 'root' or 'hpss') are granted read and write access to these resource; 2) HPSS administrators should be aware that machines, possibly owned or managed by other groups, which are added to the SAN to facilitate the use of SAN3P transfers will have access to all data on disk and tape resources. If those systems are compromised, or there are individuals authorized for systems privileges on those particular machines, but not necessarily authorized for HPSS administrative access, there is the potential for access and/or damage to HPSS data. These are inherent limitations of SAN implementations that have not yet been addressed by the industry and cannot be remedied by HPSS.*

- **Multiple Mover Tasks** (disk only). If ON, the Mover will allow multiple Mover tasks to access the disk device.
- **Reserve/Release** (tape only). An indication of whether a SCSI reservation is taken on the device when it's opened.

  Advice - This is useful on fibre attached tape devices to ensure that HPSS has sole control on the device. Without the reservation, it is possible for other hosts to interleave SCSI commands to the drive with those issued by HPSS. This effect could potentially lead to corruption of data.

State. A set of flags describing the current state of the device. The flag states are presented as a list of flag descriptions for each flag which is currently ON:

- Mounted - A media volume is mounted in the device.
- Open for Read - The device is open for reading.
- Open for Write - The device is open for writing.
- Busy - The device is in use.
- Error Encountered - An error was encountered on the device.
- Hit End-Of-Tape - The end of a tape was detected.
- Client Off Midblock - The device position is in the middle of a block.
- Data to be Synced - There are data that need to be flushed.

Set State. A set of flags showing device state changes which are pending. The flag states are presented as a list of flag descriptions for each flag which is currently ON. See the **State** field description above for a list of potential values.

Volume Flags. Flags which further describe that describe the volume type.

Number of Errors. The number of errors that have occurred on this device since the last time this field was cleared. Click on the **Reset** button to the right of the field to reset the count to zero.

Block Size. The size, in bytes, of each data block on this device.

Volume ID. The label (name) of the volume which is currently mounted on this device. If no volume is currently mounted, this field will be blank.

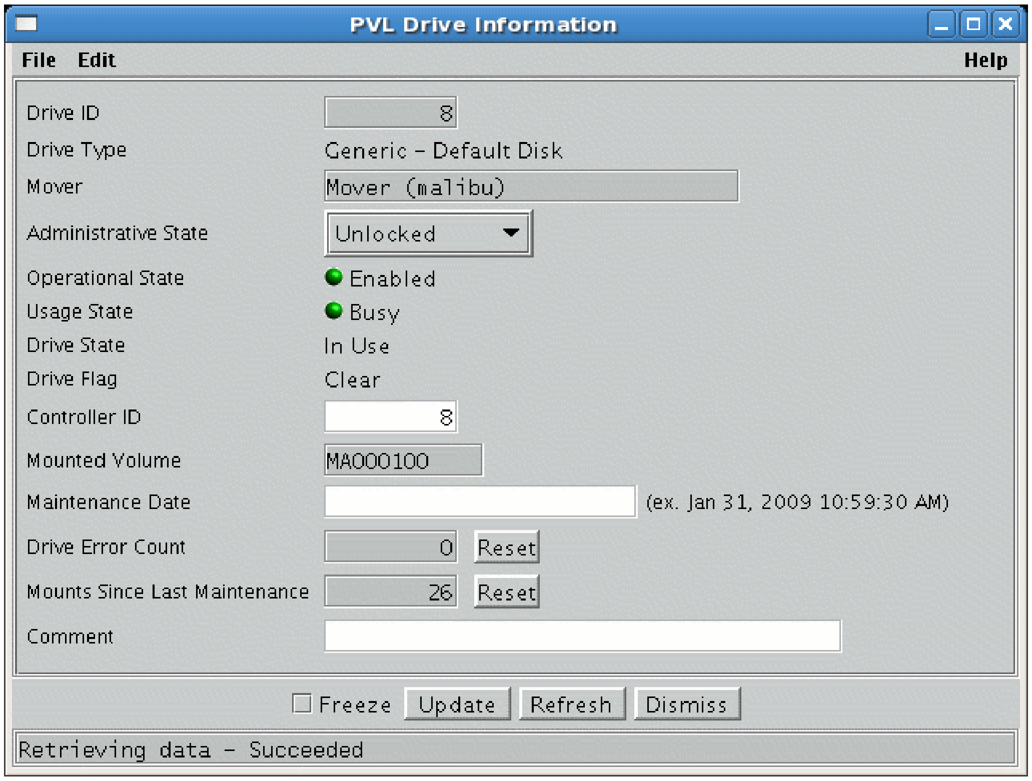
Bytes Read. The number of bytes that have been read from this device since the last time this field was cleared. Click on the **Reset** button to the right of the field to reset the count to zero.

Bytes Written. The number of bytes that have been written to this device since the last time this field was cleared. Click on the **Reset** button to the right of the field to reset the count to zero.

Operational Notes

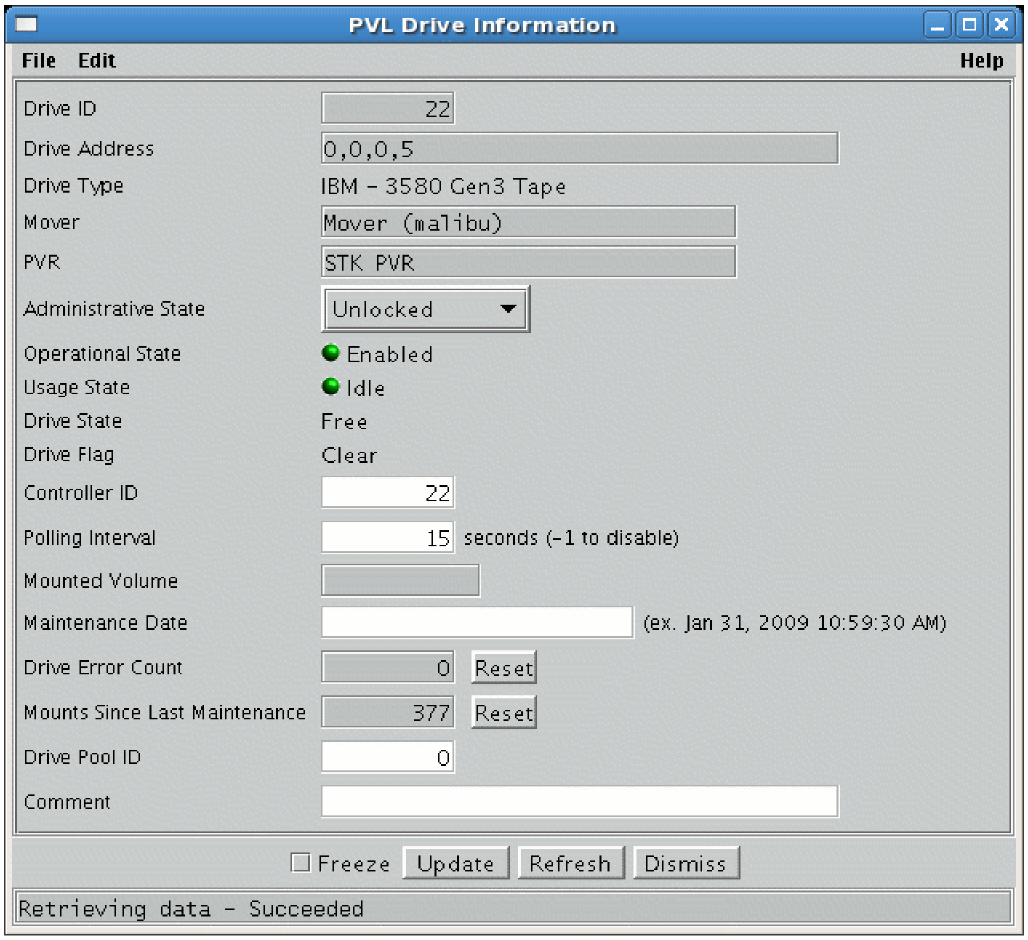
- Setting device attributes through this window will not cause the configuration metadata to be updated.
- The device statistics fields (number of bytes read, number of bytes written, number of errors) are all initialized to zero when the mover is initialized.

7.2.2. PVL Drive Information Window



Drive ID	8
Drive Type	Generic - Default Disk
Mover	Mover (malibu)
Administrative State	Unlocked
Operational State	● Enabled
Usage State	● Busy
Drive State	In Use
Drive Flag	Clear
Controller ID	8
Mounted Volume	MA000100
Maintenance Date	(ex. Jan 31, 2009 10:59:30 AM)
Drive Error Count	0 Reset
Mounts Since Last Maintenance	26 Reset
Comment	

Retrieving data - Succeeded



Drive ID	22
Drive Address	0,0,0,5
Drive Type	IBM - 3580 Gen3 Tape
Mover	Mover (malibu)
PVR	STK PVR
Administrative State	Unlocked
Operational State	● Enabled
Usage State	● Idle
Drive State	Free
Drive Flag	Clear
Controller ID	22
Polling Interval	15 seconds (-1 to disable)
Mounted Volume	
Maintenance Date	(ex. Jan 31, 2009 10:59:30 AM)
Drive Error Count	0 Reset
Mounts Since Last Maintenance	377 Reset
Drive Pool ID	0
Comment	

Retrieving data - Succeeded

This window allows you to view/update the information associated with an HPSS drive. The *PVL Drive Information* window is typically used to lock and unlock drives since newly configured drives are locked by default and must be unlocked to be used. It may also be used to determine which volume is mounted on the drive when the drive reports a mount error condition.

Any changes made to fields on this window are saved in metadata and sent directly to the PVL and thus are effective immediately. Thus the changes are dynamic.

Field Descriptions

Drive ID. The ID number unique to this drive. The same ID number also refers to the corresponding Mover device.

Drive Address (tape only). The name/address by which the drive's PVR can access it. This information is used in communications between the PVL and the PVR that controls removable media operations for the drive.

Drive Type. The HPSS drive type which describes the physical drive.

Mover. The descriptive name of the Mover which moves data to and from this drive.

PVR (tape only). The descriptive name of the PVR used to control this drive. This field is only meaningful for tape drives.

Administrative State. This field allows you to modify the state of the drive. The options are:

- Locked - Makes the drive unavailable for HPSS requests.
- Unlocked - Makes the drive available for HPSS requests.
- Mark Repaired - Tells the PVL to clear any error status for the drive. This can be useful if you think a problem has been fixed, but the PVL is unaware of it. This does not do anything, in hardware or software, to fix a problem; it only clears error indicators.

Operational State. Indicates the drive's ability to handle HPSS requests. Possible values for this field are:

- Enabled - The drive is available for HPSS requests.
- Disabled - The drive is unavailable for HPSS requests; possibly caused by setting the "Locked" Administrative State (see above) or if an error occurred when mounting or dismounting a cartridge on the drive.
- Broken - This value will appear only when the PVL is in the midst of shutting itself down because of a fatal error.
- Unknown **-** The state of the device is not known to SSM; this is usually caused by the PVL being down or disconnected from SSM.

Usage State. Indicates the state of the PVL's control over the drive. Possible values for this field are:

- Active - The drive is in use.
- Idle - The drive is not in use.
- Busy – The drive is busy.
- Unknown – The state is not known to SSM.

Drive State. The current state of the drive as reported by the PVL. Possible values for this field:

- Clear - The drive has been created, but the associated Mover and PVR (if tape) haven't successfully been notified.
- In Use - A volume is currently mounted or in the process of being mounted on the drive.
- Free - The drive is unoccupied.
- Dismount Pending - The drive is in the process of dismounting a volume.
- Deleted – The drive has been deleted (metadata no longer exists).

Drive Flag. This field describes possible actions that the PVL needs to perform for the drive. Possible flags are:

- Clear - Nothing needs to be done.
- Modified - Set by SSM when the drive has been modified. It was added for a future feature; currently unsupported.

- PVR/Mover Notify Pending - The PVL needs to notify the associated PVR and Mover that the drive has been created or deleted.
- PVR Notify Pending - The PVL needs to notify the associated PVR that the drive has been created or deleted.
- Mover Notify Pending - The PVL needs to notify the associated Mover that the drive has been created or deleted.
- Abort PVR/Mover Notify – The PVL is aborting a pending notification.

Controller ID. An indication of the adapter/bus that is in the data path to the device. This field is used to attempt to mount individual volumes of a set of striped media on different controllers, when possible, to prevent contention on the data path.

Polling Interval (tape only). The number of seconds to wait between polling requests performed by the PVL to determine if a tape or disk is present in the drive. The value for drives located within robotic libraries should be set higher than for manually mounted drives, since the PVL will query the drive after notification from the PVR that a cartridge has been mounted on it. It is recommended that this field be set to 15 seconds for operator-mounted tape drives and 900 (15 minutes) for library-managed tape drives. Use -1 to disable polling. Values of 0 to 14 are not valid.

Mounted Volume. The volume that is currently mounted on this drive.

Maintenance Date. The time and date when the drive last had maintenance performed on it. This date must be entered manually and is provided for administrative information purposes only.

Drive Error Count. This field records the number of consecutive Drive Errors (set via the **Retry Mount Time Limit** mechanism) which have occurred on the drive. When the number of errors equal or exceed the PVR **Drive Error Limit**, the drive will automatically be locked by the PVL. The field can be reset to zero administratively using the **Reset** button. A successful mount or PVL recycle will also zero the field.

Mounts Since Last Maintenance. The number of times a volume has been mounted on this drive since the last maintenance. Clicking on the **Reset** button to the right of this field will reset the value to zero.

Drive Pool ID (tape only). If non-zero positive integer value, the **Drive Pool ID** will restrict this drive to only be scheduled for tape read requests specifying this value. The default **Drive Pool ID** is 0 to indicate that the drive is not restricted to a particular drive pool. Negative values are not allowed. Changing the **Drive Pool ID** in this window can be performed while HPSS servers are up; however, the tape drive must be locked and free. See Section 7.3: *Drive Pools* on page 218.

Comment. This field provides a 128 character buffer in the PVL drive metadata which gives the administrator the opportunity to associate miscellaneous text with a device/drive. For example, a site may want to place a comment in this field that the drive is out of service or being used by another system, etc.

Related Information

Section 16.1.2: *Tape Management* on page 368

7.3. Drive Pools

HPSS provides HPSS end clients the ability to direct tape read I/O requests to a predefined group of tape drives referred to as a Drive Pool. This ability helps HPSS administrators manage tape drive scheduling and thus availability. To do this, the HPSS administrator will need to:

- Associate tape drives to a specific drive pool by configuring the HPSS tape drives with a non-zero positive integer **Drive Pool ID**.
- Modify the end client to dictate that their read request be serviced by tape drives from this particular Drive Pool.

7.3.1. Tape Drive Configuration

To associate HPSS tape drives to a specific drive pool, an HPSS administrator will need to configure HPSS tape drives with a non-zero positive integer **Drive Pool ID**. The **Drive Pool ID** can be set at configuration time via the *Tape Device Configuration* window (Section 7.1: *Configure a New Device & Drive* on page 196); it can be dynamically modified during runtime via the *PVL Drive Information* window (Section 7.2.2: *PVL Drive Information Window* on page 214) as long as the tape drive is locked and free. To change the value during downtime, you must use the *Tape Device Configuration* window and take down the HPSS servers associated with the tape drive: Mover, PVL and PVR. A **Drive Pool ID** of zero indicates the default Drive Pool.

7.3.2. Client Application Tape Read Requests

To make use of the Drive Pool feature, HPSS end client applications will need to be modified to dictate that their read request be serviced by tape drives from this particular Drive Pool. This ability is only available via the hpss_ReadList() API. The **Drive Pool ID** must be specified in the *ReqSpecInfo* structure of the I/O Descriptor (IOD) passed to hpss_ReadList. This is accomplished by specifying HPSS_IOD_SUBFUNC_DRIVE_POOL for the *SubFunction* and the **Drive Pool ID** for the *Argument* in the *ReqSpecInfo* structure of the IOD. Refer to the *HPSS Programmer's Reference - I/O Supplement* for more information.

7.3.3. Drive Pool Considerations

If HPSS receives an invalid **Drive Pool ID** or if the requested **Drive Pool ID** does not have the appropriate drive type to handle the request, then the HPSS Client API will return an error to the client for retry.

Since the **Drive Pool ID** is only used for read requests, all tape writes and tape migrations will use the default **Drive Pool ID**. It is recommended that at least one tape drive remain in the default drive pool at all times in order to satisfy these tape write requests.

Drive pool scheduling may not occur immediately since it could be the case that the drive was already scheduled before the drive was locked and the **Drive Pool ID** was changed.

If a tape is being read and a request to read the same tape arrives within 45 seconds after the first request completes, the Core Server will issue the I/O to the existing cartridge/drive. Once the drive transitions to the Deferred Dismount state, a second request with a different **Drive Pool ID** will result in a dismount and mount to a drive with the assigned **Drive Pool ID**.

Clients specifying different **Drive Pool ID**s for accesses to files from the same tape (or set of tape) may result in more tape mount/unmount activity, depending upon timing. This is another reason why the drive pool processing is restricted to tape read access only.

All mounts for striped tape volumes will be serviced from the same Drive Pool.

7.4. Changing Device and Drive State

The administrative state of a device or drive can be set to Unlocked or Locked**.** This controls whether HPSS can access the drive. Changing the state of a device or drive can be accomplished via the *Devices and Drives* list window.

Notice that there are two sets of Lock, Unlock and Mark Repaired button groups on the *Devices and Drives* window. The first group is titled **Device Administration** and the second group is titled **Drive Administration**. The first group is working at the Mover device level and the second group is working at the PVL drive level.

Locking **devices** serves no functional purpose. When the device is locked, the information is not transferred to the PVL and thus the PVL will continue to schedule the associated PVL drive. It is also not transferred to the Core Server (CS) where the CS will continue to read and write the Volume (see **VV Condition** in the *Core Server Volume Disk/Tape* window) that is mounted in the drive.

When a **drive** is locked, the PVL will no longer schedule the PVL drive. When locking a tape drive due to a cartridge problem (e.g. stuck tape), it is beneficial to also cancel the PVL Job associated with the cartridge. Additionally, locking the drive is not transferred to the Core Server (CS), therefore the CS will continue to read and write the Volume (see **VV Condition** in the *Core Server Volume Disk/Tape* window) unless you change the **VV Condition** to DOWN (or EOM) which is advised while diagnosing the stuck tape.

7.4.1. Unlocking a Drive

The Administrative State of the drive must be unlocked to allow HPSS to make use of a device/drive. Before unlocking a drive, ensure that its hardware is functional and is fully configured into its host's operating system and into HPSS. Configuration of an HPSS drive is described in Section 7.1: *Configure a New Device & Drive* on page 196.

From the *Devices and Drives* window (Section 7.1.1 *Devices and Drives Window* on page 202), select the desired device/drive entries and then click on the Unlock button from the Drive button group. The drive can also be unlocked by bringing up the *PVL Drive Information* window and set its Administrative State to Unlocked.

7.4.2. Locking a Drive

To lock a drive, go to the *Devices and Drives* window (Section 7.1.1 *Devices and Drives Window* on page 202), select the desired device/drive entries and then click on the Lock button from the Drive button group. The drive can also be locked by bringing up the *PVL Drive Information* window and set its Administrative State to Locked.

Locking a drive disallows its use by HPSS. Changing a drive state to locked will ensure that the drive will not be used for new mounts.

Always lock the PVL drive instead of the Mover device. Mover devices do not usually need to be locked or unlocked, and the capability to lock and unlock them is provided for troubleshooting purposes only.

Locking a tape drive will not affect an active job which has a volume mounted. Once that job has completed, the drive will be dismounted and no further mounts will take place. This may be useful when preventative maintenance is required for an operating drive.

Locking a disk drive has little effect since disks are logically mounted when the PVL initializes and are not usually unmounted; however, a disk drive must be in the locked state to be deleted.

7.4.3. Repairing the State of a Device or Drive

A drive can enter an error or suspect state as reported by the PVL, Mover, or both. After a drive has entered one of these abnormal states, it can be repaired to return it to a normal state.

From the *Devices and Drives* window (Section 7.1.1 on page 202), select the desired device/drive entries and then click on the Mark Repaired button appropriate for that device/drive type. Another way to change a device/drive state to "repaired" is to bring up the *PVL Drive Information* window or the *Mover Device Information* window and changing the Administrative State to Mark Repaired.

Repairing the state of a device/drive is only an instruction to the server to reset the displayed device/drive state value. It does not correct any underlying problems that might still exist. Rather, it is a means by which the administrator can notify the server that the underlying problem has been addressed outside of the server. It is used for problems that the server cannot fix by itself such as a drive being offline or a tape getting stuck in a drive.

7.4.4. Resetting Drive Statistics

Both the PVL and Mover maintain statistics related to a drive. The PVL maintains an error count as well as a count of the number of times a cartridge is mounted. The Mover maintains the count of the number of errors encountered and the number of bytes read and written on the drive. These values can all be reset to zero by the Reset button located next to the statistic field on the appropriate window.

Chapter 8. Volume and Storage Management

This chapter describes the procedures for adding, removing, monitoring, and managing storage space in the HPSS system.

The basic unit of storage which can be added to the HPSS system is the volume. Before volumes can be added to HPSS, the underlying configuration structures must be created to support them. These structures include:

- Storage classes, hierarchies, classes of service, migration policies, purge policies, and, optionally, file families. See Chapter 6 *Storage Configuration* for the procedures for creating these structures.
- Device configurations. See Chapter 7 *Devices and Drive management* for the procedures for creating these structures.

8.1. Adding Storage Space

Adding storage space to the HPSS system means creating new virtual volumes and adding them to a storage class. Creating new virtual volumes in a storage class is a two-step process:

1. **Import Volumes**. In the first step, the physical volumes that will be gathered into the new Core Server virtual volumes are imported by the PVL.
2. **Create Resources.** In the second step, metadata for the new volumes is created by the Core Server; this process is called creating storage resources.

8.1.1. Importing Volumes into HPSS

Importing a physical volume (PV) into HPSS is the process of adding a tape cartridge or disk volume to the PVL and labeling it with an HPSS volume label. Importing also makes the volume known to the PVR and PVL, and creates the appropriate metadata records. Cleaning cartridges are managed by the tape library directly and should not be imported into the HPSS system.

To import a tape cartridge into HPSS, the administrator must be familiar with the operation of the PVR to which the cartridge will be added because the physical cartridge injection process differs among PVRs.

- **Operator** - No cartridge injection step is necessary (or possible).
- **3494** - When importing new cartridges into HPSS, the cartridges must be entered into the IBM robot before any HPSS import operations are performed. Cartridges placed in the convenience I/O port will automatically be injected by the robot.
- **AML** - When importing new cartridges into HPSS, the cartridges must be entered into the AML Insert/Eject/Foreign (I/E/F) port before any HPSS import operations are performed. The HPSS AML PVR moves the cartridges into the towers or shelves before importing into HPSS. Users can then use the HPSS *Import Tape Volumes* window to import the tapes into HPSS; users do not need to issue the 'dasadmin insert' command prior to importing. See the ADIC *AML Administrator's Guide* for procedures to enter cartridges into the I/E/F bins.

  Users must configure the Insert/Eject ports for the AML PVR using the configuration files

/var/hpss/etc/AML_EjectPort.conf and /var/hpss/etc/AML_InsertPort.conf. The AML robot can have multiple insert and eject ports, which have the capability to handle different media types. These two configuration files in conjunction with the AMU AMS configuration files specify which Insert/Eject areas or bins the tapes should be placed in for insertion into the archive and the area to which they are ejected when the HPSS export command is used.

The AML PVR is supported by special bid only.

- **LTO** - When importing new cartridges into HPSS, the cartridges must be entered into the IBM 3584 input/output (I/O) station before any HPSS import operations are performed. The HPSS LTO PVR moves the cartridges into an available slot while importing into HPSS.

  Tapes that are entered into the library without an HPSS import step or tapes that are not removed after an HPSS export should not affect HPSS in any way; HPSS will continue to function as if these tapes did not exist in the library. However, tapes inserted into the I/O ports will not be filed until an import is begun.

- **SCSI** – When importing new cartridges into HPSS, the cartridges must be entered into an input/output (I/O) slot before any HPSS import operations are performed. The HPSS SCSI PVR moves the cartridges into an available slot while importing into HPSS.
- **STK** - When importing new cartridges into HPSS, the cartridges must be entered into the STK robot before any HPSS import operations are performed. See the *STK Automated Cartridge System Library Software (ACSLS) System Administrator's Guide* for procedures to enter cartridges.

  Cartridges may be entered into the STK silo through various means, including:

  - Using the enter command of ACSLS (StorageTek's robot software) and putting the cartridge(s) into the Cartridge Access Port (CAP).
  - Loading a cartridge into a CAP operating in ACSLS's "hot enter" mode.
  - Bulk loading a robot and then performing an ACSLS audit of the slots that have been filled.
  - STK supports cartridges without bar code external labels. These cartridges can be entered into the robot using the 'venter' command at the ACSLS console. These cartridges are fully supported by HPSS, but the robot will eject the cartridges if they are scanned during an audit.

We strongly recommend that tape cartridges be imported into a robot that verifies external labels before mounting in a drive. This will ensure that a blank tape being imported has the appropriate external label before it is labeled. Once labeled, a cartridge can be moved to another PVR, such as an operator (hand) mounted PVR.

If labels are written on tape volumes prior to importing those volumes into HPSS, then care should be taken that the label has a format that HPSS can process. In particular, the first four characters of the OwnerID field should be "HPSS" and the last two characters of this fourteen character field should be "00". HPSS uses the last two characters to indicate the side number of the cartridge in anticipation of supporting cartridges that have multiple physical sides - e.g., optical disk cartridges that contain an A and a B side. If these two characters are not set correctly, then the import will fail because the side information contained in the internal label will not match the side information passed to PVL in the Import request. If the start of the OwnerID field is not “HPSS”, then the volume will be imported as a Foreign Label volume and the side will be set to zero.

When importing a non-removable disk volume into HPSS, the raw disk must already be defined in the host system. The import labels the volumes with HPSS labels.

The volumes can be imported into HPSS using the *Import Disk Volumes* window (Section 8.1.1.3 on page 230) or the *Import Tape Volumes* window in the following section. Once the volumes are imported, the HPSS storage space can be created using the *Create Disk Resources* or *Create Tape Resources* window.

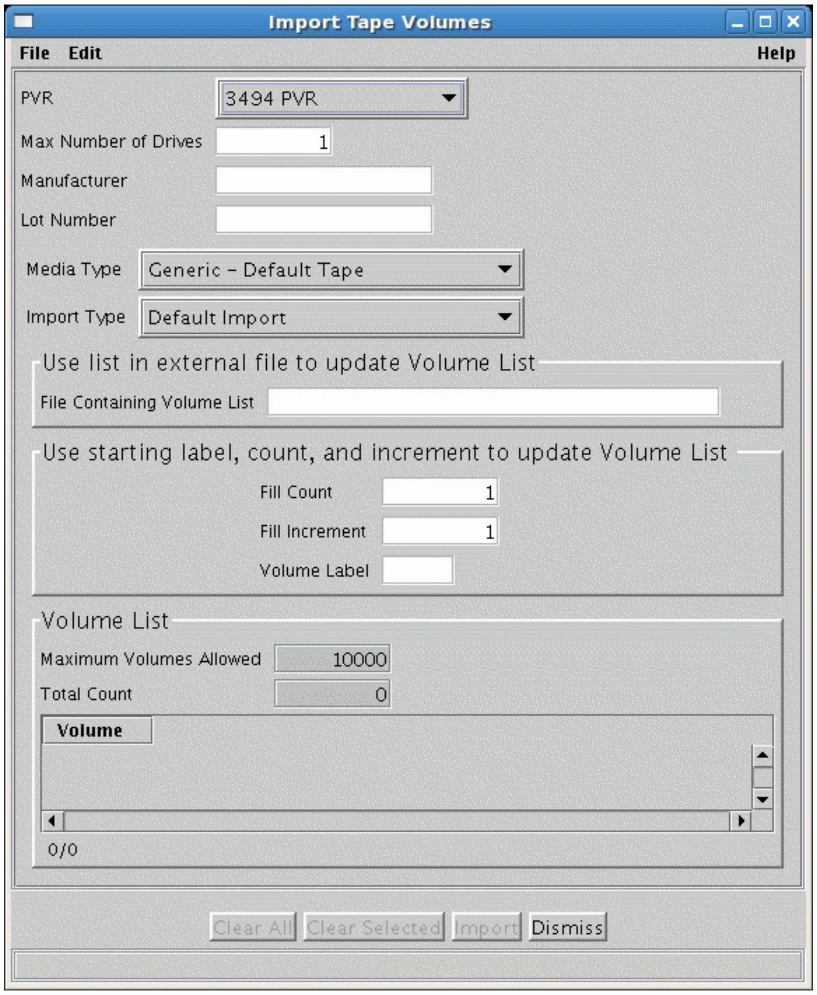
The import step must be done for both disk and tape volumes. Ensure that the PVL, PVR, and appropriate Movers are up and running before initiating a requesting for the import operation. Also, ensure that SSM is connected to the PVL.

When importing tapes, the maximum number of tape drives used by the import process is controlled by the **Max Number of Drives** field in the *Import Tape Volumes* window. Each tape must be mounted and its label must be written or checked. The PVL will attempt to evenly distribute the work across the specified number of tape drives. The administrator should consider the current and expected demand for tape drives of the type needed for the import, when setting this field.

When imports to an operator PVR are performed, requests to mount the tape cartridges are serialized. This is necessary to ensure proper labeling of unlabeled tapes. Any unlabeled cartridge mounted in a drive under control of the operator will be assumed to be the requested cartridge. Because any tape that is mounted will be labeled without machine checking of the external label, this operation is not without risk. It is strongly recommended that labeling of cartridges be performed by a robot that can verify the external label of a cartridge, if possible.

When SSM completes the import request, it reports the number of volumes imported in the window status field. Any error causes SSM to stop processing the request. If the reported number is less than the number requested, you should retry the import request after first addressing the reason for the initial failure. For the retry, the original list of volumes to import can be re-used because the PVL will skip volumes that have already been imported.

8.1.1.1. Import Tape Volumes Window


Import Tape Volumes

File Edit Help

PVR: 3494 PVR

Max Number of Drives: 1

Manufacturer:

Lot Number:

Media Type: Generic - Default Tape

Import Type: Default Import

Use list in external file to update Volume List

File Containing Volume List:

Use starting label, count, and increment to update Volume List

Fill Count: 1

Fill Increment: 1

Volume Label:

Volume List

Maximum Volumes Allowed: 10000

Total Count: 0

Volume

0/0

This window allows the user to import tape volumes into the HPSS system, making them known to the PVL server. To make them known to the Core Server so they can be used by HPSS, storage resources must then be created for the volumes via the *Create Tape Resources* window.

When the **Max Number of Drives** is set to 1, the volumes are processed one at a time in sequence. If an error occurs, processing stops. The window completion message will report the number of successful imports, from which the volume causing the failure can be found. For example, if you requested import of 100 volumes, and the completion message showed that 37 imports were successful, then the 38th volume in the Volume List caused the failure.

When the **Max Number of Drives** is greater than 1, the SSM System Manager effectively starts multiple import jobs (equal to **Max Number of Drives**) which run at the same time. Each job independently processes part of the Volume List, and stops when its part of the list is done or when an error occurs. The window completion message shows only the total of all successful imports from all of these jobs. It is therefore impossible to tell from this number which volume caused the failure. In this case it is important to review the *Alarms and Events* window to get more information on the failures.

If you request the import of a volume which was previously imported, the System Manager counts this as a successful import and goes on to the next volume in the list. This makes it easy to restart a partially completed import (after fixing the cause of the error which terminated the first request) by clicking the Import button again. There is no need to remove from the list the volumes which were imported successfully.

The list of the volumes to be imported may be constructed in any of three ways. Each volume name may be typed in one at a time, or a list of volume names may be automatically generated from a single entry, or a list of volume names may be specified from an input file. Volume names are not typed directly into the Volume List at the bottom of the window.

Any of the three entry methods may be repeated multiple times on the same window to add additional volumes to the list. All three entry methods or any combination of them may be used in succession on the same window.

To add a single volume name to the Volume List, set the **Fill Count** and the **Fill Increment** each to 1 and type the volume name into the **Volume Label** field. The volume name will be added to the Volume List.

To automatically generate a list of volume names in the Volume List, set the **Fill Count** to the desired number of volumes. Set the **Fill Increment** to the number by which each automatically generated label should differ from the previous one. Then type the starting volume name into the **Volume Label** field. The specified number of volume names will be added to the Volume List, each one larger than the previous entry by the specified **Fill Increment**.

Example:

```
Fill Count = 6
Fill Increment = 10
Volume Label = "AA0070"

Labels automatically inserted into Volume List:
"AA0070"
"AA0080"
"AA0090"
"AA0100"
"AA0110"
"AA0120"
```

When an addition produces overflow in a column, numerical columns are carried over properly to alphabetic columns and vice versa.

Example:

```
Fill Count = 6
Fill Increment = 2000
Volume Label= "AA7329"

Labels automatically inserted into Volume List:
"AA7329"
"AA9329"
"AB1329"
“AB3329”
```

```
"AB5329"
"AB7329"
```

The filling will not occur and an error will be displayed if the specified values would generate an invalid volume label (e.g., one greater than zzz999).

To specify a list of volumes from a file, create a file containing the name of each volume to be imported on a separate line. Volume names must be six alphanumeric characters. No other characters are allowed on the line. The file must be accessible from the host on which the hpssgui is executing. Enter the name of the file in the **File Containing Volume List** field. The volume names from the file will be added to the Volume List.

Field Descriptions

PVR. The descriptive name of the PVR into which the tape will be imported.

Max Number of Drives. The maximum number of tape drives to use when running the tape imports. Valid values are any positive integer value, up to the number of drives available, to a maximum of 256. The default value is 1.

Manufacturer. The maker of the media to be imported. Any character string is valid. The field may be left blank.

Lot Number. The manufacturer's lot number for the media. Any character string is valid. The field may be left blank.

Media Type. The type of media to be imported.

Import Type. The type of import to be performed. Valid values are Default, Scratch, and Overwrite. The default value is Default. Specifying Scratch Import type will usually cause an HPSS label to be written onto the media, potentially causing any data already on the media to be lost. Specifying Overwrite Import type will cause the HPSS label to be re-written, if an existing HPSS or Foreign label already exists (provided it is for the same Volume ID – this allows tapes to be re-written in newer drives (e.g., tapes originally created on a 3590 drive to be reused on a 3590E drive). Based on the **Import** Type, the import request will be processed depending on how the media is currently labeled. See Section 8.1.1.2: *Selecting Import Type for Tape Cartridges* on page 229 for more information on selecting the appropriate **Import Type**.

File Containing Volume List. The name of an external file (accessible from the host on which the hpssgui is executing) containing a list of volume labels to be added to the end of the Volume List.

Fill Count. The number of labels to be added to the Volume List when a value is typed into the **Volume Label** field. Valid values are positive integers greater than or equal to 1. The default value is 1. If the value specified for **Fill Count** would cause the list to grow beyond the Maximum Volumes Allowed, an error message is displayed (see **Maximum Volumes Allowed** below).

Fill Increment. This field determines how each new volume label is generated. The **Fill Increment** is the number by which each automatically generated label will differ from the previous one when a value is typed into the **Volume Label** field and the **Fill Count** is greater than 1. Valid values are any positive integer. The default value is 1. The **Fill Increment** is added to the least significant part of a volume label to generate a new label.

Volume Label. The six-character alpha-numeric label of a volume to be imported. The label is added to the end of the Volume list. If **Fill Count** is greater than 1, multiple labels are generated using the entered label as a starting point.

Maximum Volumes Allowed. The maximum number of volume labels that will fit in the Volume List. The value is 10,000 and is set by SSM. This field is non-editable.

Total Count. The total number of tapes to be imported. This is an informational field reflecting the number of volume names generated in the Volume List and is not directly editable.

Volume List. A list of volume labels specifying the volumes to be imported. You cannot enter labels directly into this list, but must construct it in one of the three ways described above. Use the scrollbar to move up and down the list of labels.

You can select one or more labels from the list for the purpose of deleting them from the list (see **Clear Selected** below). To select one label, click on it with the mouse; the selection will be highlighted. To select a range, select one label; then select a second label while holding down the Shift key. The selected labels, and all labels between the two, will be highlighted. You may also hold down the Control key while selecting, to select or deselect individual labels without affecting any of the others.

Buttons

Clear All. Clears the Volume List, and resets **Fill Count** and **Fill Increment** to 1.

Clear Selected. If one or more volumes are selected (highlighted) in the Volume list, clicking on this button will remove them from the list. Note that this does not actually delete anything from HPSS.

Import. Begins the volume import using the displayed data. A start message is displayed on the message line at the bottom of the window, and all window features are disabled, except for the **Dismiss** button; this prevents you from modifying any data on the window while the import is in progress.

It is a good idea to have the *Alarms and Events* window open while import proceeds, so that all relevant log messages can be viewed. For Tape imports, it is also helpful to have the *Tape Mount Requests* window open, where you can see mount requests for each volume come and go as they are imported.

When the import is finished, a completion message is displayed and the window features are sensitized again. You may dismiss the window before completion; however, completion messages will be displayed in a pop-up window. At this point you can begin entering data for another import, or you can dismiss the window.

8.1.1.2. Selecting Import Type for Tape Cartridges

The following table lists information for selecting tape import types.

Any existing data on a volume imported into HPSS will be overwritten and subsequently lost!

Table 3. Tape Import Types

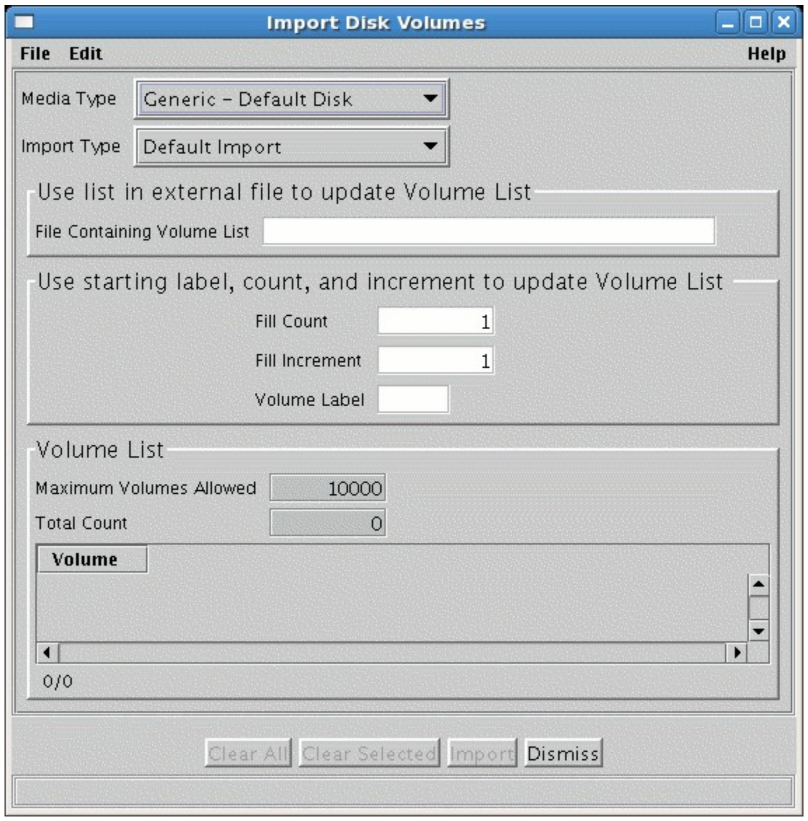
Current Tape Label	Default Import	Overwrite Import	Scratch Import
A UniTree label (an ANSI label without a trailing tape mark) with a correct Volume ID.	Tape Imported	Label Written, Tape Imported	Tape Imported

An ANSI (non-UniTree) or HPSS label with a correct Volume ID (the Volume ID on the label is as expected by HPSS)	Tape Imported	Label Written, Tape Imported	Tape Imported
An ANSI or HPSS label with an incorrect Volume ID (the Volume ID on the label is different from the Volume ID expected by HPSS)	Tape Not Imported	Tape Not Imported	Tape Not Imported
Random data (e.g., a tar file)	Tape Not Imported	Label Written, Tape Imported	Label Written, Tape Imported
No data (two tapemarks at the start of tape)	Label Written, Tape Imported	Label Written, Tape Imported	Label Written, Tape Imported
Unreadable (e.g., some brand new tapes or a degaussed tape; also possibly a tape written at a different density)	Tape Not Imported	Label Written, Tape Imported	Label Written, Tape Imported

HPSS always attempts to read a label at the beginning of a tape when performing an import. Some tape drives and device drivers will report errors when asked to read unreadable tapes (as described above). If this happens, manually write two tape marks at the start of the tape and retry the import. Most UNIX systems provide the mt command, which can be used to write tape marks.

An HPSS label is basically just an ANSI label. A few characters are changed to identify the tape as having been labeled by HPSS. Both label types are supported by HPSS; tapes already beginning with an ANSI label are not relabeled unless the overwrite import option is selected and volume labels match.

8.1.1.3. Import Disk Volumes Window


Import Disk Volumes
File Edit Help
Media Type: Generic - Default Disk
Import Type: Default Import
Use list in external file to update Volume List
File Containing Volume List
Use starting label, count, and increment to update Volume List
Fill Count: 1
Fill Increment: 1
Volume Label
Volume List
Maximum Volumes Allowed: 10000
Total Count: 0
Volume
0/0
Clear All
Clear Selected
Import
Dismiss

This window allows the user to import disk volumes into the HPSS system, making them known to the PVL and PVR servers. To make them known to the Core Server so they can be used, they must be created via the Core Server's *Create Disk Resources* window.

The SSM System Manager processes the volumes one at a time in sequence. If it encounters an error, it stops and returns. The window completion message will report the number of successful imports, from which the volume causing the failure can be found. For example, if you requested import of 100 volumes, and the completion message showed that 37 imports were successful, then the 38th volume in the Volume List caused the failure.

If you request the import of a volume which was previously imported, the System Manager counts this as a successful import and goes on to the next volume. This makes it easy to restart a partially completed import (after fixing the cause of the error which terminated the first request) by clicking the Import button again. There is no need to remove the already imported volumes from the list.

The list of the volumes to be imported may be constructed in any of three ways. Each volume name may be typed in one at a time, or a list of volume names may be automatically generated from a single entry, or a list of volume names may be specified from an input file. Volume names are not typed directly into the Volume List at the bottom of the window.

Any of the three entry methods may be repeated multiple times on the same window to add additional volumes to the list. All three entry methods or any combination of them may be used in succession on the same window.

To add a single volume name to the Volume List, set the **Fill Count** and the **Fill Increment** each to 1 and type the volume name into the **Volume Label** field. The volume name will be added to the Volume

List.

To automatically generate a list of volume names in the Volume List, set the **Fill Count** to the desired number of volumes. Set the **Fill Increment** to the number by which each automatically generated label should differ from the previous one. Then type the starting volume name into the **Volume Label** field. The specified number of volume names will be added to the Volume List, each one larger than the previous entry by the specified **Fill Increment**.

Example:

```
Fill Count = 6
Fill Increment = 10
Volume Label = "AA0070"

Labels automatically inserted into Volume List:
"AA0070"
"AA0080"
"AA0090"
"AA0100"
"AA0110"
"AA0120"
```

When an addition produces overflow in a column, numerical columns are carried over properly to alphabetic columns and vice versa.

Example:

```
Fill Count = 6
Fill Increment = 2000
Volume Label= "AA7329"

Labels automatically inserted into Volume List:
"AA7329"
"AA9329"
"AB1329"
"AB3329"
"AB5329"
"AB7329"
```

The filling will not occur and an error will be displayed if the specified values would generate an invalid volume label (e.g., one greater than zzz999).

To specify a list of volumes from a file, create a file containing the name of each volume to be imported on a separate line. Volume names must be six alphanumeric characters. No other characters are allowed on the line. The file must be accessible from the host on which the hpssgui is executing. Enter the name of the file in the **File Containing Volume List** field. The volume names from the file will be added to the Volume List.

Field Descriptions

Media Type. The type of media to be imported.

Import Type. The type of import to be performed. Valid values are Default, Scratch and Overwrite .

Based on the Import Type, the import request will be processed depending on how the media is currently labeled. See Section 8.1.1.4: *Selecting Import Type for Disk Volumes* on page 234 for more information on selecting the appropriate Import Type.

File Containing Volume List. The name of an external file containing a list of volume labels to be added to the end of the Volume List.

Fill Count. The number of labels to be added to the Volume List when a value is typed into the Volume Label field. Valid values are positive integers greater than or equal to 1. The default value is 1. If the value specified for **Fill Count** would cause the list to grow beyond the Maximum Volumes Allowed, an error message is displayed (see **Maximum Volumes Allowed** below).

Fill Increment. This field determines how each new volume label is generated. The **Fill Increment** is the number by which each automatically generated label will differ from the previous one when a value is typed into the Volume Label field and the **Fill Count** is greater than 1. Valid values are any positive integer. The default value is 1. The **Fill Increment** is added to the least significant part of a volume label to generate a new label.

Volume Label. The six-character alpha-numeric label of a volume to be imported. The label is added to the end of the Volume list. If **Fill Count** is greater than 1, multiple labels are generated using the entered label as a starting point.

Maximum Volumes Allowed. The maximum number of volume labels that will fit in the Volume List. The value is 10,000 and is set by SSM. This field is non-editable.

Total Count. The total number of disks to be imported. This is an informational field reflecting the number of volume names generated in the Volume List and is not directly editable.

Volume List. A list of volume labels specifying the volumes to be imported. You cannot enter labels directly into this list, but must construct it using one of the three ways described above. Use the scrollbar to move up and down the list of labels.

You can select one or more labels from the list for the purpose of deleting them from the list (see **Clear Selected** below). To select one label, click on it with the mouse; the selection will be highlighted. To select a range, select one label; then select a second label while holding down the Shift key. The selected labels, and all labels between the two, will be highlighted. You may also hold down the Control key while selecting, to select or deselect individual labels without affecting any of the others.

Buttons

Clear All. Clears the Volume List, and resets **Fill Count** and **Fill Increment** to 1.

Clear Selected. If one or more volumes are selected (highlighted) in the Volume list, clicking on this button will remove them from the list. Note that this does not actually delete anything from HPSS.

Import. Begins the volume import using the displayed data. A start message is displayed on the message line at the bottom of the window, and all window features are disabled, except for the **Dismiss** button; this prevents you from modifying any data on the window while the import is in progress.

It is a good idea to have the *Alarms and Events* window open while import proceeds, so that all relevant log messages can be viewed.

When the import is finished, a completion message is displayed and the window features are sensitized again. You may dismiss the window before completion; however, completion messages will be displayed in a pop-up window. At this point you can begin entering data for another import, or you can dismiss the window.

8.1.1.4. Selecting Import Type for Disk Volumes

The following table lists information for selecting disk import types.

Table 4. Disk Import Types

Current Disk Label	Default Import	Overwrite Import	Scratch Import
An ANSI or HPSS label with a correct Volume ID (the Volume ID on the label is as expected by HPSS)	Disk Imported	Label Written, Disk Imported	Disk Imported
An ANSI or HPSS label with an incorrect Volume ID (the Volume ID on the label is different from the Volume ID expected by HPSS)	Disk Not Imported	Disk Not Imported	Disk Not Imported
No label	Label Written, Disk Imported	Label Written, Disk Imported	Label Written, Disk Imported

8.1.2. Creating Storage Resources

Creating resources is the process of creating Core Server metadata to manage the space on HPSS volumes. The volumes must be imported into HPSS (see Section 8.1.1: *Importing Volumes into HPSS* on page 223) before resources may be created on them.

Before storage resources can be created, the administrator must determine which storage sub-system will receive the new storage, and which storage class they will be placed in. The choice of storage sub-system determines the Core Server that will be used.

Ensure that the PVL and the appropriate Core Server are running before initiating a Create Storage Resources request. Also, ensure that SSM is connected to the appropriate Core Server.

The selected Core Server will create a number of HPSS virtual volumes. All new tape virtual volumes will have identical characteristics (type, estimated size, stripe size, stripe width, block size, blocks between tape marks, etc.). When creating disk storage resources, the administrator may override the default PV Size and enter a value specific to the circumstance.

Virtual volumes are created by the Core Server by grouping physical volumes together in groups of N, where N is the Stripe Width of the resulting virtual volume. The physical volumes are arranged into virtual volumes according to the table in the resource creation window, which the administrator should modify as desired before clicking the Create button. Every physical volume is placed into a virtual volume before it is usable by HPSS, even if there is only one physical volume per virtual volume.

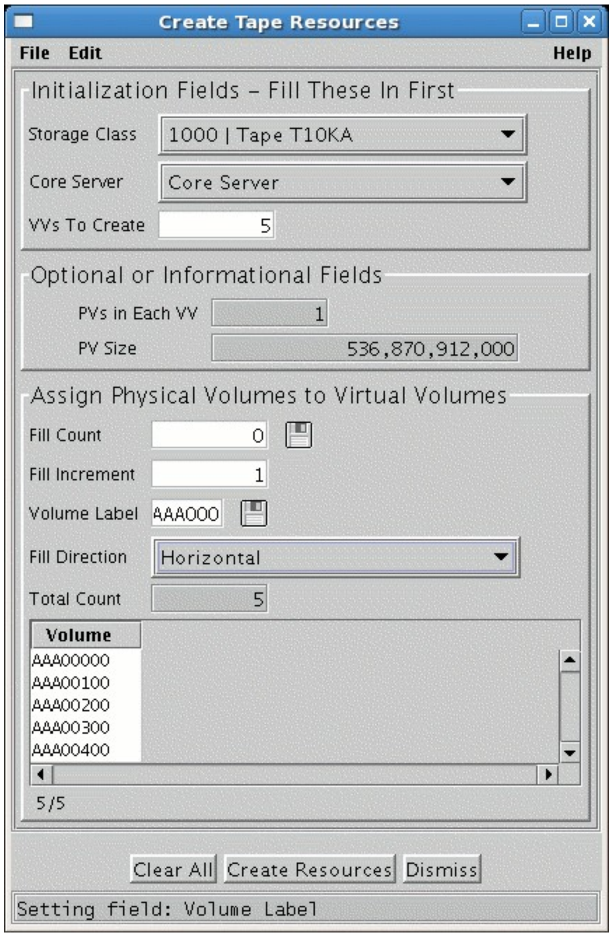
It is important to understand that the Storage Class characteristics in effect when volumes are created are assigned to the new volumes. Once volumes are created, changes to the Storage Class have no effect on the characteristics of existing volumes. The Storage Class characteristics (block sizes, stripe widths, etc.)

are copied to the volumes' metadata and become a permanent part of the definition of the volumes.

The Core Server creates the necessary metadata structures for each of the new virtual volumes. Each new virtual volume is immediately available for use.

Note that new tape resources are not assigned to a file family. Tapes are assigned to file families from the unused tapes in the storage class as they are needed to satisfy requests to write to tape in a given family. Tapes cannot be pre-assigned to file families.

8.1.2.1. Create Tape Resources Window


Create Tape Resources
File
Edit
Help
Initialization Fields - Fill These In First
Optional or Informational Fields
Assign Physical Volumes to Virtual Volumes
Clear All
Create Resources
Dismiss

Storage Class: 1000 | Tape T10KA
Core Server: Core Server
VVs To Create: 5
PVs in Each VV: 1
PV Size: 536,870,912,000
Fill Count: 0
Fill Increment: 1
Volume Label: AAA000
Fill Direction: Horizontal
Total Count: 5

Volume
AAA00000
AAA00100
AAA00200
AAA00300
AAA00400

5/5

Setting field: Volume Label

This window is used to create tape storage resources, tape virtual volumes, in a Core Server. The tapes must first be imported to the appropriate PVL.

The names of the volumes may be entered into the window one at a time, or a list of volume names may be automatically generated from a single entry. Volume names are not entered directly into the Volume List at the bottom of the window but are typed into the **Volume Label** field.

To add a single volume name to the Volume List, set the **Fill Count** and the **Fill Increment** each to 1 and type the volume name into the **Volume Label** field. The volume name will be added to the Volume

List.

To automatically generate a list of volume names in the Volume List, set the **Fill Count** to the desired number of volumes. Set the **Fill Increment** to the number by which each automatically generated label should differ from the previous one. Then type the starting volume name into the **Volume Label** field. The specified number of volume names will be added to the Volume List, each one larger than the previous entry by the specified **Fill Increment**.

Example:

```
Fill Count = 6
Fill Increment = 10
Volume Label = "AA0070"

Labels automatically inserted into Volume List table:
"AA0070"
"AA0080"
"AA0090"
"AA0100"
"AA0110"
"AA0120"
```

When an addition produces overflow in a column, numerical columns are carried over properly to alphabetic columns and vice versa.

Example:

```
Fill Count = 6
Fill Increment = 2000
Volume Label= "AA7329"

Labels automatically inserted into Volume List table:
"AA7329"
"AA9329"
"AB1329"
“AB3329”
“AB5329”
“AB7329”
```

Once the Core Server completes the Create request, SSM reports the number of physical volumes and virtual volumes created in the window status field.

Field Descriptions

Initialization Fields

Storage Class. The name of the Storage Class whose characteristics will be used to create the requested virtual volume(s). The stripe width of this storage class will determine the number of columns in the Volume List table at the bottom of the window.

Core Server. The name of the Core Server that will process the create request and in which the tape storage resources will be created. The Storage Class field must be filled in before selecting a Core

Server.

VVs To Create. The number of virtual volumes to be created. This value determines the number of rows in the Volume List table at the bottom of the window.

Optional or Informational Fields

PVs in Each VV. The Stripe Width of the selected Storage Class. This field may not be changed.

PV Size. The size, in bytes, of each physical volume used. The default value is the value from the storage class. For tapes, this is an estimate and is informational only.

Assign Physical Volumes to Virtual Volumes

Fill Count. The number of labels to be added to the Volume List when a value is typed into the **Volume Label** field. Valid values are positive integers greater than or equal to 1 up to the number of drives available. The default value is 1. If the list fills before the **Fill Count** is exhausted, filling stops and a message is displayed.

Fill Increment. This field determines how each new volume label is generated. The **Fill Increment** is the number by which each automatically generated label will differ from the previous one when a value is in the **Volume Label** field and the **Fill Count** is greater than 1. Valid values are positive integers up to the number of available drives. The default value is 1. A volume label is six alphanumeric characters. The **Fill Increment** is added to the least significant portion of a volume label to generate a new label.

Volume Label. The next label that will be generated and inserted into the table. The labels are generated when the field loses focus. Select **Fill Count**, **Fill Increment**, and **Fill Direction** before entering a value in this field.

Fill Direction. Determines which direction filling takes place in the table. By default, **Fill Direction** is Horizontal. Select the desired value for this field before entering a **Volume Label**.

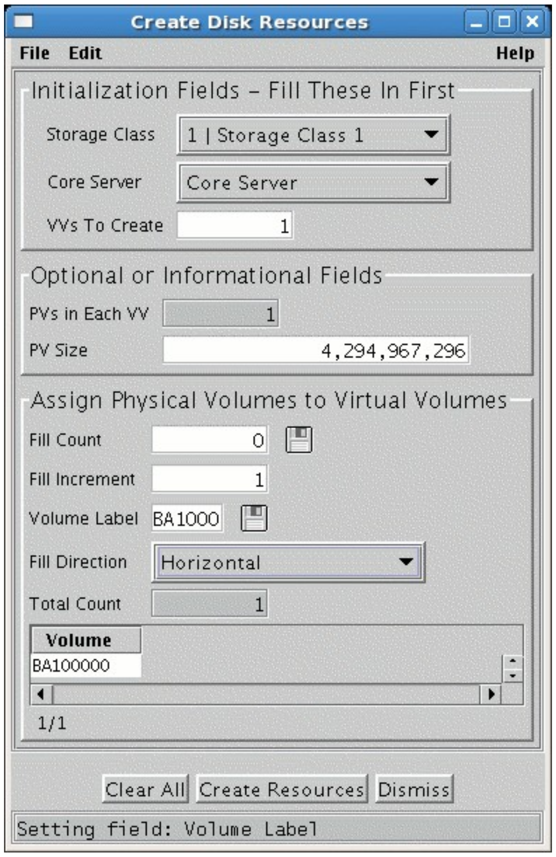
Total Count. The total number of physical volumes in the generated Volume List. This is an informational field and not directly enterable.

Volume List. A list of volume labels specifying the volumes used to create resources. You cannot enter labels directly into this list but must instead use the **Fill Count**, **Fill Increment**, and **Volume Label** fields. Use the scrollbar to navigate the list of labels.

Clear All. This button clears the Volume list and **Volume Label** field.

Create Resources. Begins the process of creating HPSS storage resources using the displayed data. A start message is displayed on the status bar, and all window features are disabled, except for the **Dismiss** button. This prevents the user from modifying any data on the window while the create request is in progress.

8.1.2.2. Create Disk Resources Window



Initialization Fields - Fill These In First

Storage Class	1 | Storage Class 1
Core Server	Core Server
VVs To Create	1

Optional or Informational Fields

PVs in Each VV	1
PV Size	4,294,967,296

Assign Physical Volumes to Virtual Volumes

Fill Count	0
Fill Increment	1
Volume Label	BA1000
Fill Direction	Horizontal
Total Count	1

Volume
BA100000

1/1

Setting field: Volume Label

This window is used to create disk storage resources, disk virtual volumes, in a Core Server. The disks must first be imported to the appropriate PVL.

The names of the volumes may be entered into the window one at a time, or a list of volume names may be automatically generated from a single entry. Volume names are not entered directly into the Volume List at the bottom of the window but are typed into the **Volume Label** field.

To add a single volume name to the Volume List, set the **Fill Count** and the **Fill Increment** each to 1 and type the volume name into the **Volume Label** field. The volume name will be added to the Volume List.

To automatically generate a list of volume names in the Volume List, set the **Fill Count** to the desired number of volumes. Set the **Fill Increment** to the number by which each automatically generated label should differ from the previous one. Then type the starting volume name into the **Volume Label** field. The specified number of volume names will be added to the Volume List, each one larger than the previous entry by the specified **Fill Increment**.

Example:

```
Fill Count = 6
Fill Increment = 10
Volume Label = "AA0070"

Labels automatically inserted into Volume List table:
"AA0070"
```

```
"AA0080"
"AA0090"
"AA0100"
"AA0110"
"AA0120"
```

When an addition produces overflow in a column, numerical columns are carried over properly to alphabetic columns and vice versa.

Example:

```
Fill Count = 6
Fill Increment = 2000
Volume Label= "AA7329"

Labels automatically inserted into Volume List table:
"AA7329"
"AA9329"
"AB1329"
"AB3329"
"AB5329"
"AB7329"
```

Once the Core Server completes the Create request, SSM reports the number of physical volumes and virtual volumes created in the window status field.

Field Descriptions

Initialization Fields

Storage Class. The name of the Storage Class whose characteristics will be used to create the requested virtual volume(s). The stripe width of this storage class will determine the number of columns in the Volume List table at the bottom of the window.

Core Server. The name of the Core Server that will process the create request and in which the disk storage resources will be created. The Storage Class field must be filled in before selecting a Core Server.

VVs To Create. The number of virtual volumes to be created. This value determines the number of rows in the Volume List table at the bottom of the window.

Optional or Informational Fields

PVs in Each VV. The Stripe Width of the selected Storage Class. This field may not be changed.

PV Size. The size, in bytes, of each physical volume used. The default value is the value from the storage class. For disks, it should be changed on this window to the value that matches the actual value of each disk being imported.

Assign Physical Volumes to Virtual Volumes

Fill Count. The number of labels to be added to the Volume List when a value is typed into the **Volume Label** field. Valid values are positive integers greater than or equal to 1 up to the number of drives available. The default value is 1. If the list fills before the **Fill Count** is exhausted, filling stops and a message is displayed.

Fill Increment. This field determines how each new volume label is generated. The **Fill Increment** is the number by which each automatically generated label will differ from the previous one when a value is in the **Volume Label** field and the **Fill Count** is greater than 1. Valid values are positive integers up to the number of available drives. The default value is 1. A volume label is six alphanumeric characters. The **Fill Increment** is added to the least significant portion of a volume label to generate a new label.

Volume Label. The next label that will be generated and inserted into the table. The labels are generated when the field loses focus. Select **Fill Count**, **Fill Increment**, and **Fill Direction** before entering a value in this field.

Fill Direction. Determines which direction filling takes place in the table. By default, **Fill Direction** is Horizontal. Select the desired value for this field before entering a **Volume Label**.

Total Count. The total number of physical volumes in the generated Volume List. This is an informational field and not directly enterable.

Volume List. A list of volume labels specifying the volumes used to create resources. You cannot enter labels directly into this list but must instead use the **Fill Count**, **Fill Increment**, and **Volume Label** fields. Use the scrollbar to navigate the list of labels.

Clear All. This button clears the Volume list and **Volume Label** field.

Create Resources. Begins the process of creating HPSS storage resources using the displayed data. A start message is displayed on the status bar, and all window features are disabled, except for the **Dismiss** button. This prevents the user from modifying any data on the window while the create request is in progress.

8.2. Removing Storage Space

Removing storage space from the HPSS system is a two step process:

1. **Delete Resources**. In the first step, the Core Server virtual volume metadata is deleted; this process is called deleting storage resources.
2. **Export Volumes**. In the second step, the physical volumes that belonged to the virtual volumes are exported from the PVL.

The second step, exporting the volumes from the PVL, is not always necessary. Once the resources are deleted in step 1, new storage resources can be created on the volumes without the need to export the volumes from the system and reimport them. Exporting the volumes is necessary only if the volumes are not going to be reused in the HPSS system, or if for some reason they must be relabeled. The site may wish to export the volumes if they are to be imported into a different PVR in the system, but this function can also be accomplished with the Move Cartridges operation.

8.2.1. Deleting Storage Resources

Storage resources can be removed from HPSS volumes by the use of the *Delete Resources* window or the **remove** utility.

Once resources have been deleted, the volumes may be reused or removed from the HPSS system. To reuse the volumes, create new storage resources on them using the *Create Resources* window. To remove them entirely from the HPSS system, export them from the PVL using the *Export Volumes* window.

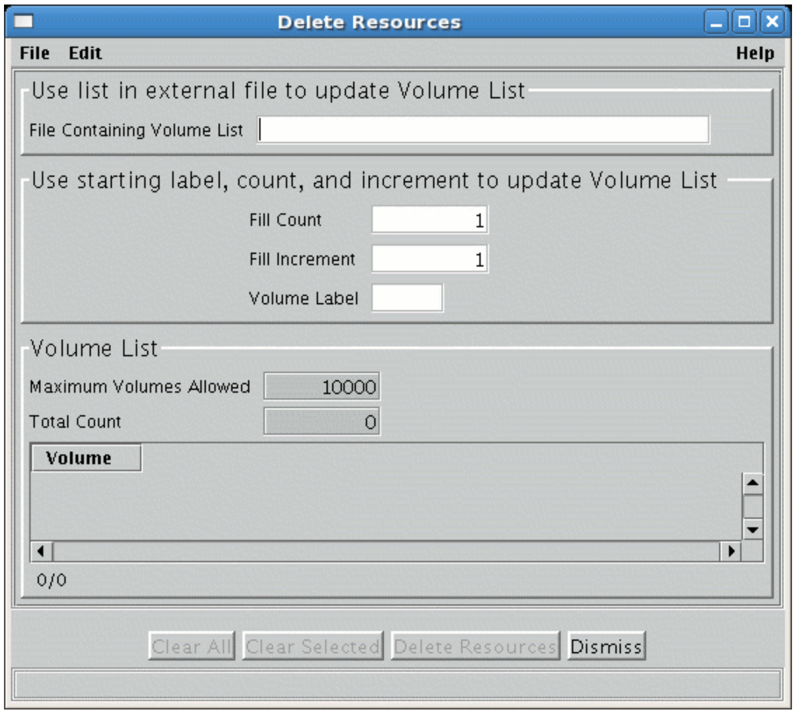
8.2.1.1. Rules for Deleting Resources

Volumes on which resources are to be deleted must be empty. There must be no storage segments associated with the volumes. The *Core Server Disk Volume* or *Tape Volume* window must show a zero in the Number of Active Segments field, and the VV Condition must be EMPTY. If any storage segments remain on a volume, the resource deletion operation will fail. You can use the volume **repack** features of HPSS to empty disk and tape volumes so that their resources may be deleted.

The Delete Resources operation may be performed on either retired or non-retired volumes.

Successful deletions result in the removal of the Core Server's metadata records from the appropriate subsystem. The volume is then no longer usable for storage by HPSS, but it is still recognized by the PVL.

8.2.1.2. Delete Resources Window



Use list in external file to update Volume List

File Containing Volume List

Use starting label, count, and increment to update Volume List

Fill Count: 1

Fill Increment: 1

Volume Label

Volume List

Maximum Volumes Allowed: 10000

Total Count: 0

Volume

0/0

This window is used to delete empty or unused storage resources (i.e., virtual volumes) from a Core Server. The volumes must have no storage segments on them when they are removed.

The list of the volumes to be deleted may be constructed in any of three ways. Each volume name may be typed in one at a time, or a list of volume names may be automatically generated from a single entry, or a list of volume names may be specified from an input file. Volume names are not typed directly into the

Volume List at the bottom of the window.

Any of the three entry methods may be repeated multiple times on the same window to add additional volumes to the list. All three entry methods or any combination of them may be used in succession on the same window.

To add a single volume name to the Volume List, set the **Fill Count** and the **Fill Increment** each to 1 and type the volume name into the **Volume Label** field. The volume name will be added to the Volume List.

To automatically generate a list of volume names in the Volume List, set the **Fill Count** to the desired number of volumes. Set the **Fill Increment** to the number by which each automatically generated label should differ from the previous one. Then type the starting volume name into the **Volume Label** field. The specified number of volume names will be added to the Volume List, each one larger than the previous entry by the specified **Fill Increment**.

Example:

```
Fill Count = 6
Fill Increment = 10
Volume Label = "AA0070"

Labels automatically inserted into Volume List:
"AA0070"
"AA0080"
"AA0090"
"AA0100"
"AA0110"
"AA0120"
```

When an addition produces overflow in a column, numerical columns are carried over properly to alphabetic columns and vice versa.

Example:

```
Fill Count = 6
Fill Increment = 2000
Volume Label= "AA7329"

Labels automatically inserted into Volume List:
"AA7329"
"AA9329"
"AB1329"
“AB3329”
“AB5329”
“AB7329”
```

The filling will not occur and an error will be displayed if the specified values would generate an invalid volume label (e.g., one greater than zzz999).

To specify a list of volumes from a file, create a file containing the name of each volume to be deleted on a separate line. Volume names must be six alphanumeric characters. No other characters are allowed on the line. The file must be accessible from the host on which the hpssgui is executing. Enter the name of the file in the **File Containing Volume List** field. The volume names from the file will be added to the Volume List.

Field Descriptions

File Containing Volume List. The name of an external file containing a list of volume labels to be added to the end of the Volume List.

Fill Count. The number of volume labels to be added to the end of the Volume List when the **Volume Label** field is next modified.

Fill Increment. In a multiple-label fill, this field determines how each new PV label is generated from the previous one. Each new volume label entered in the table is the previous entry incremented by this amount.

Volume Label. The label to be added to the end of the PV list. If **Fill Count** is greater than 1, this will be the first label of a fill sequence. The label must be exactly six characters long. Entering a value in this field causes the table entries to be generated.

Maximum Volumes Allowed. The maximum number of volume labels that will fit in the Volume List. This field is informational and not enterable.

Total Count. The total number of volumes in the list to be deleted. This is the current size of the generated Volume List.

Volume List. A list of volume labels of volumes to be deleted. You cannot enter labels directly into this list but must construct it using one of the three methods described above. You can select one or more labels from the list for the purpose of removing them from the list (see **Clear Selected** below). To select one label, click on it with the mouse; the selection will be highlighted. To select a range, select one label; then select a second label while holding the Shift key. The selected labels, and all labels between them, will be highlighted. You may also hold down the Control key while selecting, to select or de-select individual labels without affecting any other selections.

Clear All. Clears the Volume list and resets **Fill Count** and **Fill Increment**.

Clear Selected. If one or more volumes are selected (highlighted) in the Volume list, clicking on this button will remove them from the list. Note that this does not delete anything from HPSS.

Delete Resources. Begins the deletion of resources using the displayed data. A start message is displayed on the status bar, and all window features are disabled except for the **Dismiss** button. This prevents the user from modifying any data on the window while the delete request is in progress.

Related Information

Section 8.4: *Dealing with a Space Shortage* on page 258.

8.2.2. Exporting Volumes from HPSS

Exporting volumes from HPSS is the process of removing tape cartridges or disk volume from HPSS control.

8.2.2.1. Rules for Exporting Volumes

Tape cartridges may be physically exported from any managing robotic library. To export a tape cartridge from HPSS, the administrator must be familiar with the operation of the PVR from which the cartridge will be removed because the physical cartridge ejection process differs among the PVRs supported by HPSS:

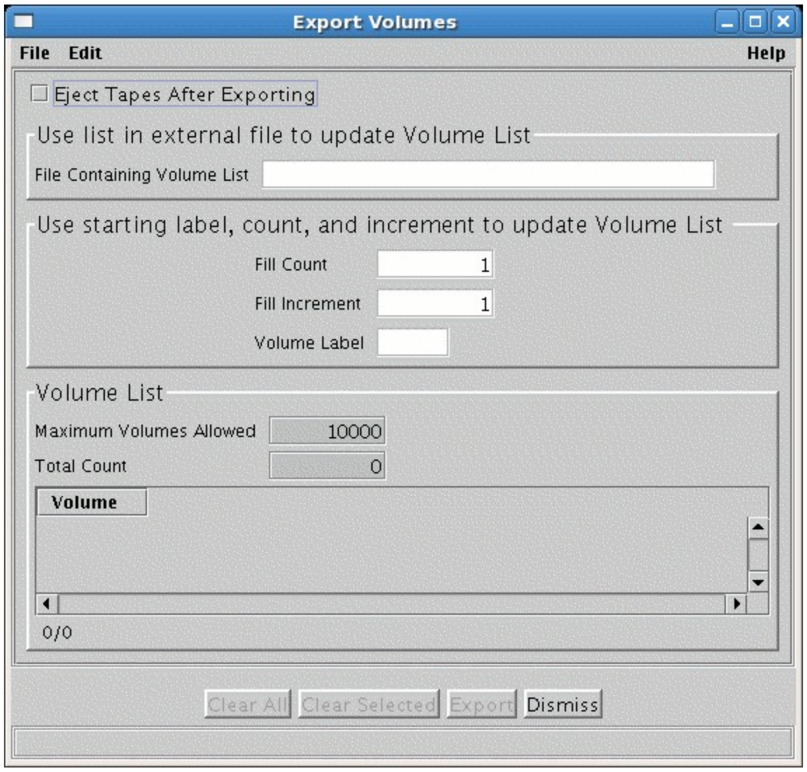
- **Operator** - No cartridge ejection step is necessary (or possible).
- **3494** - When ejecting cartridges, HPSS will send cartridges to the convenience I/O port if it is available. If the port is not available and a high capacity I/O region is defined, the cartridges will be placed in that region. If no locations are available for the cartridge to be ejected, an alarm will appear on the HPSS operator window and HPSS will retry the eject operation periodically.
- **AML** - The exported cartridges will be placed in the E/I/F ports and can then be manually removed.
- **LTO** - When an HPSS tape export command is issued, the cartridges will be placed in the input/output (I/O) stations and can then be manually removed.
- **SCSI** - When an HPSS tape export command is issued, the cartridges will be placed in the input/output (I/O) slot and can then be manually removed.
- **STK** - The exported cartridges will be placed in the CAP and can then be manually removed. The export operation is NOT completed until cartridges are manually removed.

Only imported but unallocated volumes can be exported. Volumes allocated to a Core Server can be unallocated by deleting all Core Server metadata that describe the volume using the procedure described in Section 8.2.1: *Deleting Storage Resources* on page 240.

When exporting a removable cartridge from a robot, the cartridges will normally be sent to the convenience I/O port if it is available (this will not be done if the Export Without Eject option is used). If no locations are available for the cartridge to be ejected, an alarm will appear on the *Alarms and Events* window and HPSS will periodically retry the eject operation.

Use the *Export Volumes* window in the following section to perform the actual export operation.

8.2.2.2. Export Volumes Window



☐ Eject Tapes After Exporting

Use list in external file to update Volume List

File Containing Volume List

Use starting label, count, and increment to update Volume List

Fill Count 1

Fill Increment 1

Volume Label

Volume List

Maximum Volumes Allowed 10000

Total Count 0

0/0

This window allows you to export tape and disk volumes from the HPSS system. Exporting a volume is equivalent to telling HPSS that the volume no longer exists. Before volumes can be exported, the Core Server storage resources that describe the volumes must be deleted using the procedure described in Section 8.2.1: *Deleting Storage Resources* on page 240.

The list of the volumes to be exported may be constructed in any of three ways. Each volume name may be typed in one at a time, or a list of volume names may be automatically generated from a single entry, or a list of volume names may be specified from an input file. Volume names are not entered directly into the Volume List at the bottom of the window.

Any of the three entry methods may be repeated multiple times on the same window to add additional volumes to the list. All three entry methods or any combination of them may be used in succession on the same window.

To add a single volume name to the Volume List, set the **Fill Count** and the **Fill Increment** fields each to 1 and type the volume name into the **Volume Label** field. The volume name will be added to the Volume List.

To automatically generate a list of volume names in the Volume List, set the **Fill Count** to the desired number of volumes. Set the **Fill Increment** to the number by which each automatically generated label should differ from the previous one. Then type the starting volume name into the **Volume Label** field. The specified number of volume names will be added to the Volume List, each one larger than the previous entry by the specified **Fill Increment**.

Example:

```
Fill Count = 6
Fill Increment = 10
Volume Label = "AA0070"

Labels automatically inserted into Volume List:
"AA0070"
"AA0080"
"AA0090"
"AA0100"
"AA0110"
"AA0120"
```

When an addition produces overflow in a column, numerical columns are carried over properly to alphabetic columns and vice versa.

Example:

```
Fill Count = 6
Fill Increment = 2000
Volume Label= "AA7329"

Labels automatically inserted into Volume List:
"AA7329"
"AA9329"
"AB1329"
“AB3329”
“AB5329”
“AB7329”
```

The filling will not occur and an error will be displayed if the specified values would generate an invalid volume label (e.g., one greater than zzz999).

To specify a list of volumes from a file, create a file containing the name of each volume to be exported on a separate line. Volume names must be six alphanumeric characters. No other characters are allowed on the line. The file must be accessible from the host on which the hpssgui is executing. Enter the name of the file in the **File Containing Volume List** field. The volume names from the file will be added to the Volume List.

The entire list of volumes is sent to the System Manager as a single export request. The System Manager processes them one at a time, and if it encounters an error, it stops and returns. In this case, the window status line will report the number of successful exports, and the volume name that caused the failure. Since the System Manager does not ignore volumes that have already been exported, but instead considers them failed exports, you cannot retry an export by resubmitting the same list of volumes. You will first have to remove the successfully exported volumes from the list. The **Clear Selected** button can be useful for this.

Once the export is completed, cartridges residing in robotic tape systems will be ejected by the controlling PVR (if **Eject Tapes after Exporting** is selected) and should be removed from the access port of the robot.

Field Descriptions

Eject Tapes After Exporting. If this checkbox is selected, the exported tape volumes will also be ejected from the PVR.

File Containing Volume List. The name of an external file containing a list of volume labels to be added to the end of the Volume List.

Fill Count. The number of volume labels to be added to the end of the list when the **Volume Label** field is next modified. This number may be one or greater. If the list fills up before the **Fill Count** is exhausted, filling stops, and a message box is displayed (see **Maximum Volumes Allowed** below).

Fill Increment. In a multiple-label fill, this field determines how each new volume label is generated from the previous one. A volume label is six alphanumeric characters. The **Fill Increment** is added to the numerical part of a volume label to generate a new label.

Volume Label. The volume label to be added to the end of the Volume List. If **Fill Count** is greater than 1, this will be the first label of a fill sequence. The label must be exactly six characters long. Lower case characters entered into this field will be converted to upper case.

Maximum Volumes Allowed. The maximum number of volume labels that will fit in the Volume List. This value is 10,000 and is maintained by SSM. This field is non-editable.

Total Count. The total number of volumes currently in the Volume List.

Volume List. A list of volume labels specifying the volumes to be exported. You cannot enter labels directly into this list, but must construct it using one of the three ways described above. Use the scrollbar to move up and down the list of labels.

You can select one or more labels from the list for the purpose of deleting them from the list (see **Clear Selected** below).

Buttons

Clear All. Clears the Volume List and resets **Fill Count** and **Fill Increment** to 1.

Clear Selected. If one or more volumes are selected (highlighted) in the Volume List, clicking on this button will remove them from the list. Note that this does not actually delete anything from HPSS.

Export. Begins the volume export using the displayed data. A start message is displayed on the status bar at the bottom of the window, and all window features are disabled, except for the **Dismiss** button; this prevents you from modifying any data on the window while the export is in progress.

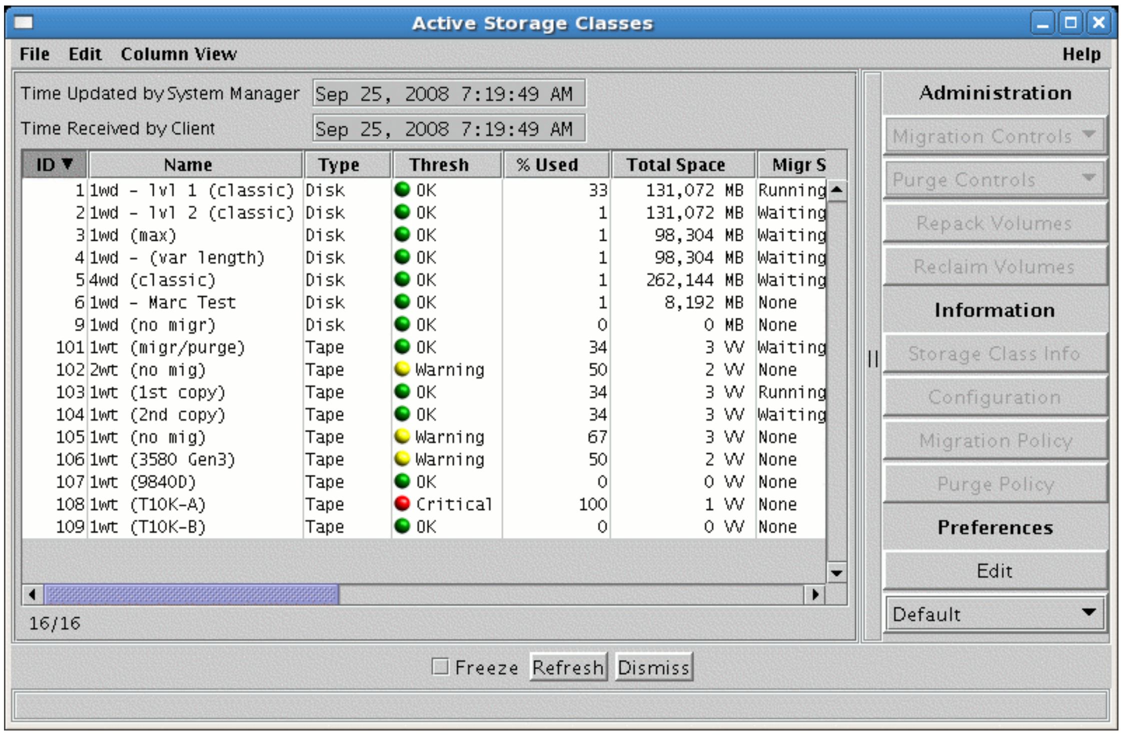
8.3. Monitoring Storage Space

One of the most important administrative tasks is to monitor storage space usage in the HPSS system. The goal is early detection of any potential space shortages so that corrective action can be taken in a timely manner. Each storage class is configured with a warning threshold and a critical threshold to inform SSM users when the free space in a storage class runs low. Warning and major alarms are sent to the *Alarms and Events* window periodically when the warning and critical thresholds are exceeded. Migration policies can be configured to automatically start migration when either threshold is exceeded.

When a storage class experiences a threshold-exceeded condition, an administrator may need to review the associated migration and purge policies for the storage class as well as the total storage space available to the storage class. The migration and purge policies may need to be modified to free up more space or to free up the space more frequently. In addition, the total storage space for the storage class may need to be reviewed to determine whether it is sufficient to accommodate the actual usage of the storage class.

Storage space in HPSS is monitored from the *Active Storage Classes* window.

8.3.1. Active Storage Classes Window



Time Updated by System Manager: Sep 25, 2008 7:19:49 AM
Time Received by Client: Sep 25, 2008 7:19:49 AM

ID ▼	Name	Type	Thresh	% Used	Total Space	Migr S
1	1wd - lvl 1 (classic)	Disk	● OK	33	131,072 MB	Running
2	1wd - lvl 2 (classic)	Disk	● OK	1	131,072 MB	Waiting
3	1wd (max)	Disk	● OK	1	98,304 MB	Waiting
4	1wd - (var length)	Disk	● OK	1	98,304 MB	Waiting
5	4wd (classic)	Disk	● OK	1	262,144 MB	Waiting
6	1wd - Marc Test	Disk	● OK	1	8,192 MB	None
9	1wd (no migr)	Disk	● OK	0	0 MB	None
101	1wt (migr/purge)	Tape	● OK	34	3 VV	Waiting
102	2wt (no mig)	Tape	● Warning	50	2 VV	None
103	1wt (1st copy)	Tape	● OK	34	3 VV	Running
104	1wt (2nd copy)	Tape	● OK	34	3 VV	Waiting
105	1wt (no mig)	Tape	● Warning	67	3 VV	None
106	1wt (3580 Gen3)	Tape	● Warning	50	2 VV	None
107	1wt (9840D)	Tape	● OK	0	0 VV	None
108	1wt (T10K-A)	Tape	● Critical	100	1 VV	None
109	1wt (T10K-B)	Tape	● OK	0	0 VV	None

16/16

The *Active Storage Classes* window allows you to view a list of all active storage classes. An active storage class is defined as a storage class configured in HPSS to which storage volumes have been assigned. Volumes are assigned to a storage class by the Create Resources operation.

The information displayed in this window is obtained from the MPS, which obtains part of it from the Core Servers.

The Migration Purge Server must be running and connected to SSM or the Active Storage Classes window will be stale. All Core Servers must be running or the storages classes managed by them will not be reported in this window.

If multiple storage subsystems exist, this window will display a list of storage classes for each subsystem. To display this window, from the *Health and Status* window, select the Monitor menu, and from that select the Storage Classes Active menu item.

The function buttons to the right of this list require that one storage class row be selected from the list.

To select a row, click on it with the mouse; the selection will be highlighted. Note that when you select a row, you are selecting a storage class within a particular storage subsystem.
See also the related window *Configured Storage Classes*, described in Section 6.1.1 on page 157. The *Configured Storage Classes* window lists all configured storage classes in the system, whether or not any storage resources have been assigned to them.

Field Descriptions

Active Storage Classes list

The **Column View** menu item may be used to determine which of the following columns are displayed in the list;

Subsystem. The name of the storage subsystem to which the storage class information applies. It is possible to have multiple rows in the table with identical storage class ID and Name columns, each for a different storage subsystem.

ID. The unique storage class ID number defined when the storage class is configured.

Name. The name of the storage class.

Type. The storage class type; possible values are "Disk" or "Tape".

Thresh. The current threshold status for the storage class. For disk, a threshold is defined as a percentage of the total bytes of storage space. When the percentage of storage space used rises above a threshold level, the threshold is exceeded. For tape, a threshold is defined as a count of free virtual volumes. When the number of free volumes drops below a threshold level, the threshold is exceeded. Possible values are:

- OK - The storage class has not reached its warning threshold.
- Stale - The storage class is active but the MPS responsible for it is shutdown, has died, or is otherwise inaccessible to SSM.
- Warning - The storage class has reached its warning threshold, but has not reached its critical threshold.
- Critical - The storage class has reached its critical threshold.

Free Space. The amount of free space left in the storage class. For disk, this is the number of free bytes. For tape, it is the number of free virtual volumes.

% Free. The Free Space value expressed as a percentage of Total Space.

Used Space. The amount of used space in the storage class. For disk, this is the number of used bytes. For tape, it is the number of used virtual volumes.

% Used. The Used Space value expressed as a percentage of Total Space.

Partially Filled VVs. The number of tapes that have been written, but have not yet reached EOM. This field will be blank for disk storage classes.

Total Space. The total amount of space in the storage class. For disk, this is the number of bytes. For tape, it is the number of virtual volumes.

Migr State. The migration state for the storage class. The Purge State resets to WAITING or NONE as appropriate when the MPS is recycled. Possible values are:

- Waiting - Migration is not taking place at this time. The start of the next migration is waiting until criteria specified in the migration policy are met.
- Running - A migration is in progress.
- None - The storage class, as configured, is not a candidate for migration.
- Suspended - Migration has been suspended.

Migr Policy. The migration policy assigned to the storage class.

Purge State. The purge state for the storage class. The Purge State resets to WAITING or NONE as appropriate when the MPS is recycled. Possible values are:

- Waiting - Purging is not taking place at this time. The start of the next purge is waiting until criteria specified in the purge policy are met.
- Running - A purge is in progress.
- None - The storage class, as configured, is not a candidate for purging.
- Suspended - Purging has been suspended.

Purge Policy. The purge policy assigned to the storage class.

Administration Buttons.

Migration Controls. This pull-down can be used to send migration commands to the MPS on the selected storage classes. This pull-down is enabled only if at least one of the selected storage classes has been assigned a migration policy. Click on it to drop down a list of control options. These are:

- **Start** - If the migration state is Waiting, this starts the migration and puts it into the Running state. This does not guarantee that any files will be migrated, only that files will be reevaluated against the appropriate migration policy and migrated as appropriate.
- **Stop** - If the migration state is Running, this stops the migration and puts it into the Waiting state.
- **Suspend** - If the migration state is Waiting or Running, this puts the migration into the Suspended state.
- **Resume** - If the migration state is Suspended, this returns it to Waiting and allows MPS to again begin scheduling migration runs.
- **Reread policy** - Tells the MPS to refresh its migration policy information by rereading the policy.

Purge Controls. This pull-down can be used to send purge commands to the MPS on the selected storage classes. This pull-down is enabled only if at least one of the selected storage classes has been assigned a purge policy. Click on it to drop down a list of control options. These are:

- **Start** - If the purge state is Waiting, this starts the purge and puts it into the Running state. This does not guarantee that any files will be purged, only that files will be reevaluated against the appropriate purge policy and purged as appropriate.
- **Stop** - If the purge state is Running, this stops the purge and puts it into the Waiting state.

- **Suspend** - If the purge state is Waiting or Running, this puts the purge into the Suspended state.
- **Resume** - If the purge state is Suspended, this returns it to Waiting and allows MPS to again begin scheduling purge runs.
- **Reread policy** - Tells the MPS to refresh its purge policy information by rereading the policy.

Repack Volumes. Opens the *Repack Virtual Volumes* window, allowing you to start the **repack** utility program on the selected storage class. See Section 8.4.3.1: *Repack Virtual Volumes Window* on page 260 for details. This button is only enabled for tape storage classes.

Reclaim Volumes. Opens the *Reclaim Virtual Volumes* window, allowing you to start the **reclaim** utility program on the selected storage class. See Section 8.4.3.2: *Reclaim Virtual Volumes Window* on page 262 for details. This button is only enabled for tape storage classes.

Information Buttons.

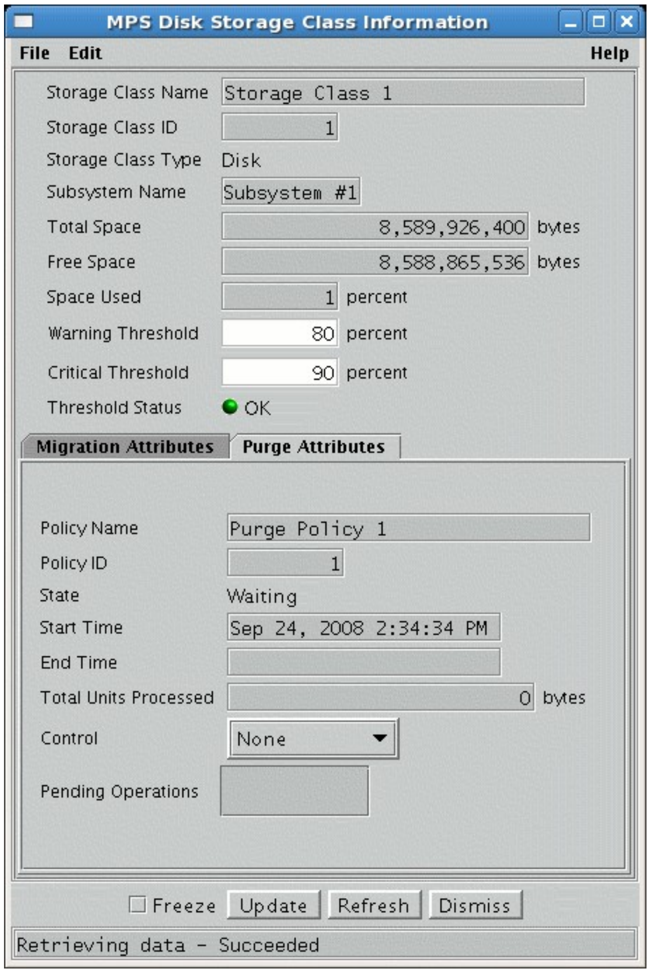
Storage Class Info. Opens the *MPS Storage Class Information* window for the selected storage class. For detailed information on usage of this window, see Section 8.3.2: *MPS Disk Storage Class Information*.

Configuration. Opens the *Storage Class Configuration* window for the selected storage class.

Migration Policy. Opens the *Migration Policy* window for the migration policy assigned to the selected storage class. If there is no assigned migration policy, the button is disabled.

Purge Policy. Opens the Purge Policy window for the purge policy assigned to the selected storage class. If there is no assigned purge policy, the button is disabled.

8.3.2. MPS Disk Storage Class Information



Storage Class Name	Storage Class 1
Storage Class ID	1
Storage Class Type	Disk
Subsystem Name	Subsystem #1
Total Space	8,589,926,400 bytes
Free Space	8,588,865,536 bytes
Space Used	1 percent
Warning Threshold	80 percent
Critical Threshold	90 percent
Threshold Status	● OK

Policy Name	Purge Policy 1
Policy ID	1
State	Waiting
Start Time	Sep 24, 2008 2:34:34 PM
End Time	
Total Units Processed	0 bytes
Control	None
Pending Operations	

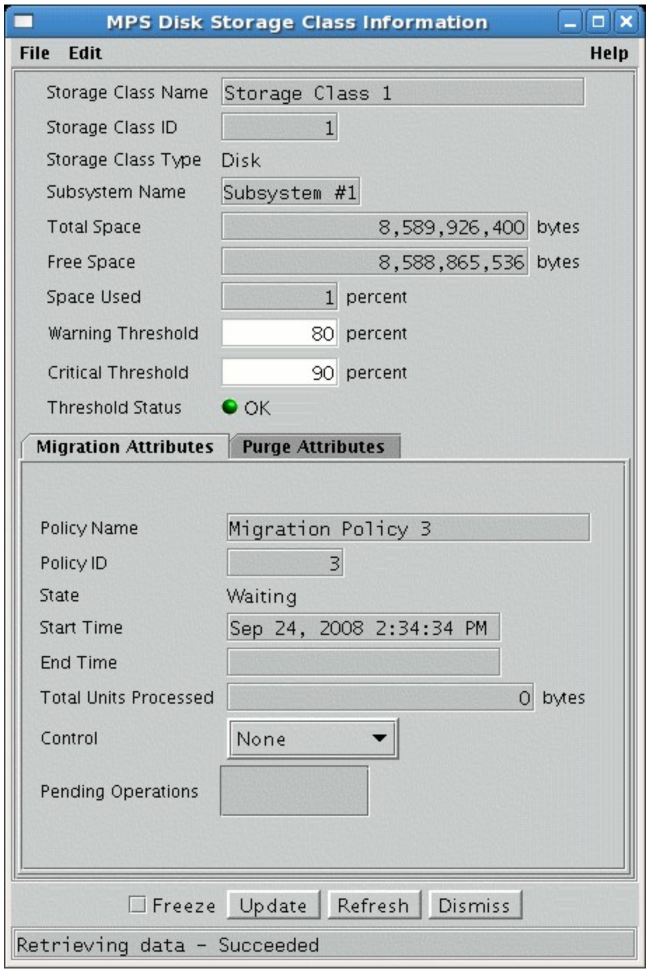
Retrieving data - Succeeded

This window allows you to view and update the information associated with an active disk storage class. It reports the storage space data as well as any exceeded thresholds. The window also reports detailed information on the migration and purge status. In addition, the window allows the SSM user to control the migration and purge process to override the associated migration and purge policies.

There are three differences between the disk and tape storage class information windows. First, the disk window reports space in terms of bytes while the tape window reports space in terms of virtual volumes (VVs). Second, the disk window reports Space Used while the tape window reports Percent Used. Third, the disk window shows purge status while the tape window does not.

Since an MPS is specific to a storage subsystem, the information in this window applies only to the volumes of the storage class which are assigned to a single storage subsystem. It is possible for multiple subsystems to use a storage class. If two of these windows are opened, each for the same storage class but for different subsystems, they will display different information.

Any changes made to fields on the window are sent directly to the MPS and are effective immediately. If the MPS is restarted, changes made through this window are lost.



Storage Class Name	Storage Class 1
Storage Class ID	1
Storage Class Type	Disk
Subsystem Name	Subsystem #1
Total Space	8,589,926,400 bytes
Free Space	8,588,865,536 bytes
Space Used	1 percent
Warning Threshold	80 percent
Critical Threshold	90 percent
Threshold Status	● OK
Policy Name	Migration Policy 3
Policy ID	3
State	Waiting
Start Time	Sep 24, 2008 2:34:34 PM
End Time	
Total Units Processed	0 bytes
Control	None
Pending Operations	

Retrieving data - Succeeded

Field Descriptions

Storage Class Name. The name assigned to this storage class.

Storage Class ID. The numeric ID of this storage class.

Storage Class Type. The class of media assigned to this storage class (tape or disk).

Subsystem Name. The name of the storage subsystem for which the storage class information is being displayed.

Total Space. For disk storage classes this reports the total capacity of the storage class in bytes.

Free Space. For disk storage classes this reports the unused capacity of the storage class in bytes.

Space Used. The percent of storage space in the storage class which is used. This is calculated from Total Space and Free Space and reported as a percentage.

Warning Threshold. For disk storage classes, this value is a percentage of total storage space. When the used space in this storage class exceeds this percent of the total space, the MPS will send a warning alarm to SSM.

Critical Threshold. For disk storage classes, this value is a percentage of total storage space. When the used space in this storage class exceeds this percent of the total space, the MPS will send a critical alarm to SSM.

Note that the disk threshold reaches the warning level when the percent of used space rises above the threshold value, while the tape threshold reaches the warning level when the free volume count drops below the threshold value.

If you wish to disable the threshold so it will not generate any alarms, set this field to 100 percent for disk, 0 volumes for tape.

Threshold Status. The current used space threshold status. Possible values are OK, Warning, and Critical.

Migration Attributes Tab. This panel contains the current migration status for this storage class.

Policy Name. The name of the migration policy assigned to this storage class.

Policy ID. The ID of the migration policy assigned to this storage class.

State. The current migration state for this storage class. Possible values are:

- **None** - This storage class, as configured, is not a candidate for migration.
- **Running** - A migration on this storage class is in progress.
- **Waiting** - Migration is not taking place at this time. The start of the next migration is waiting until criteria specified in the migration policy are met.
- **Suspended** - Migration on this storage class has been suspended by the system administrator.

Start Time. The date and time when the most recent migration run started. It may still be running.

End Time. The date and time when the last migration run completed.

Total Units Processed. The amount of space in the storage class which has been migrated during the current or or most recent migration run. For disk storage classes and for tape storage classes running the tape file migration algorithm, this is a number of bytes.

Control. This pull-down can be used to send commands to the MPS regarding migration on this storage class. Click on it to drop down a list of control options. These are:

- **None -** Nothing has been selected for the migration **Control**.
- **Start** - If the migration state is Waiting, this starts the migration and puts it into the Running state.
- **Stop** - If the migration state is Running, this stops the migration and puts it into the Waiting state.
- **Suspend** - If the migration state is Waiting or Running, this puts the migration into the

Suspended state.

- **Resume** - If the migration state is Suspended, this returns it to Waiting and allows MPS to again begin scheduling migration runs.
- **Reread policy** - Tells the MPS to refresh its migration policy information by rereading the policy.

Pending Operations. When the MPS cannot respond immediately to a Control command, a command may be saved as pending. Any such pending operations are displayed here.

Purge Attributes Tab. This panel contains the current purge status for this storage class. This tab appears only on the *MPS Disk Storage Class Information* window.

Policy Name. The name of the purge policy assigned to this storage class.

Policy ID. The ID of the purge policy assigned to this storage class.

State. The current purge state for this storage class. Possible values are:

- **None** - This storage class, as configured, is not a candidate for purging.
- **Running** - A purge on this storage class is in progress.
- **Waiting** - Purging is not taking place at this time. The start of the next purge is waiting until criteria specified in the purge policy are met.
- **Suspended** - Purging on this storage class has been suspended.

Start Time. The date and time when the most recent purge run started. It may still be running.

End Time. The date and time when the last purge run completed.

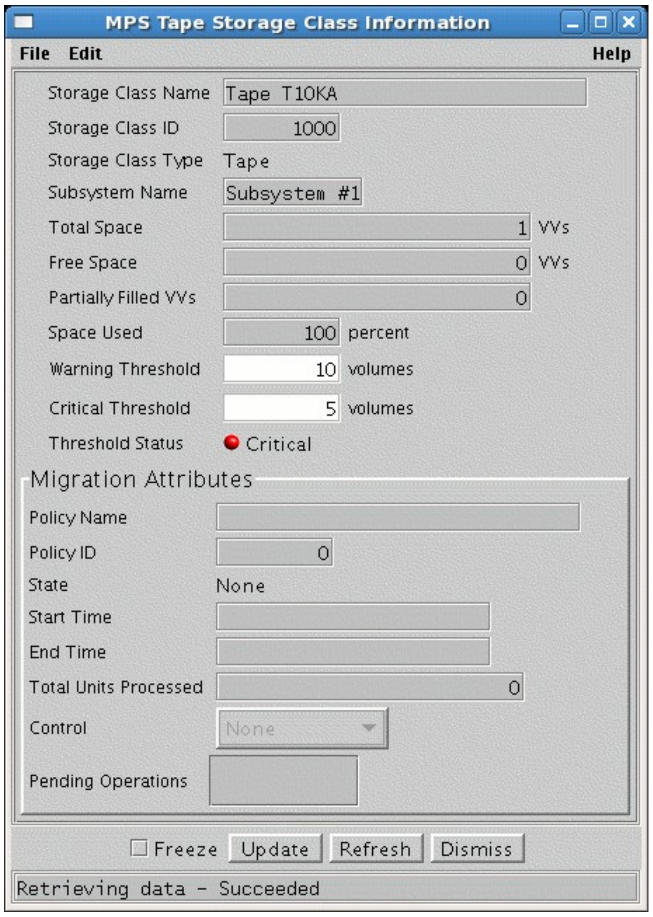
Total Units Processed. The amount of space in the storage class which has been purged, in bytes.

Control. This button can be used to send commands to the MPS regarding purging on this storage class. Click on the button to drop down a list of control options. These are:

- **None -** Nothing has been selected for the purge **Control**.
- **Start** - If the purge state is Waiting, this starts the purge and puts it into the Running state.
- **Stop** - If the purge state is Running, this stops the purge and puts it into the Waiting state.
- **Suspend** - If the purge state is Waiting or Running, this puts the purge into the Suspended state.
- **Resume** - If the purge state is Suspended, this returns it to Waiting and allows MPS to again begin scheduling purge runs.
- **Reread policy** - Tells the MPS to refresh its purge policy information by rereading the policy.

Pending Operations. When the MPS cannot respond immediately to a Control command, a command may be saved as pending. Any such pending operations are displayed here.

8.3.3. MPS Tape Storage Class Information



Storage Class Name	Tape T10KA
Storage Class ID	1000
Storage Class Type	Tape
Subsystem Name	Subsystem #1
Total Space	1 VVs
Free Space	0 VVs
Partially Filled VVs	0
Space Used	100 percent
Warning Threshold	10 volumes
Critical Threshold	5 volumes
Threshold Status	● Critical

Migration Attributes

Policy Name	
Policy ID	0
State	None
Start Time	
End Time	
Total Units Processed	0
Control	None
Pending Operations	

Retrieving data - Succeeded

This window allows you to view and update the information associated with an active tape storage class. It reports the storage space data as well as any exceeded thresholds. The window also reports detailed information on the migration status. In addition, the window allows the SSM user to control the migration process to override the associated migration policies.

There are three differences between the disk and tape storage class information windows. First, the disk window reports space in terms of bytes while the tape window reports space in terms of virtual volumes (VVs). Second, the disk window reports Space Used while the tape window reports Percent Used. Third, the disk window shows purge status while the tape window does not.

Since an MPS is specific to a storage subsystem, the information in this window applies only to the volumes of the storage class which are assigned to a single storage subsystem. It is possible for multiple subsystems to use a storage class. If two of these windows are opened, each for the same storage class but for different subsystems, they will display different information.

Any changes made to fields on the window are sent directly to the MPS and are effective immediately.

Field Descriptions

Storage Class Name. The name assigned to this storage class.

Storage Class ID. The numeric ID of this storage class.

Storage Class Type. The class of media assigned to this storage class (tape or disk).

Subsystem Name. The name of the storage subsystem for which the storage class information is being displayed.

Total Space. For tape storage classes this reports the total capacity of the storage class in virtual volumes (VVs).

Free Space. For tape storage classes this reports the number of unwritten virtual volumes (VVs) in the storage class.

Partially Filled VVs. The number of tapes that have been written, but have not yet reached EOM.

Space Used. The percent of storage space in the storage class which is used. This is calculated from Total Space and Free Space and reported as a percentage.

Warning Threshold. For tape storage classes, this value is a count of free tape volumes. When the number of free volumes in this storage class drops below this level, the MPS will send a warning alarm to SSM.

Critical Threshold. For tape storage classes, this value is a count of free tape volumes. When the number of free volumes in this storage class drops below this level, the MPS will send a critical alarm to SSM.

Note that the disk threshold reaches the warning level when the percent of used space rises above the threshold value, while the tape threshold reaches the warning level when the free volume count drops below the threshold value.

If you wish to disable the threshold so it will not generate any alarms, set this field to 100 percent for disk, 0 volumes for tape.

Threshold Status. The current used space threshold status. Possible values are OK, Warning, and Critical.

Migration Attributes. This panel contains the current migration status for this storage class.

Policy Name. The name of the migration policy assigned to this storage class.

Policy ID. The ID of the migration policy assigned to this storage class.

State. The current migration state for this storage class. Possible values are:

- **None** - This storage class, as configured, is not a candidate for migration.
- **Running** - A migration on this storage class is in progress.
- **Waiting** - Migration is not taking place at this time. The start of the next migration is waiting until criteria specified in the migration policy are met.
- **Suspended** - Migration on this storage class has been suspended by the system

administrator.

Start Time. The date and time when the most recent migration run started. It may still be running.

End Time. The date and time when the last migration run completed.

Total Units Processed. The amount of space in the storage class which has been migrated during the current or or most recent migration run. For tape storage classes running the tape volume migration algorithm, this is a number of virtual volumes (VVs).

Control. This pull-down can be used to send commands to the MPS regarding migration on this storage class. Click on it to drop down a list of control options. These are:

- **None -** Nothing has been selected for the migration **Control**.
- **Start** - If the migration state is Waiting, this starts the migration and puts it into the Running state.
- **Stop** - If the migration state is Running, this stops the migration and puts it into the Waiting state.
- **Suspend** - If the migration state is Waiting or Running, this puts the migration into the Suspended state.
- **Resume** - If the migration state is Suspended, this returns it to Waiting and allows MPS to again begin scheduling migration runs.
- **Reread policy** - Tells the MPS to refresh its migration policy information by rereading the policy.

Pending Operations. When the MPS cannot respond immediately to a Control command, a command may be saved as pending. Any such pending operations are displayed here.

8.4. Dealing with a Space Shortage

If free space warning thresholds are exceeded in a storage class, HPSS will warn the administrator with messages in the *Alarms and Events* window. Before the available storage space in a storage class drops to a level where the storage class may no longer be able to satisfy requests, efforts should be made to create additional free space. This can be accomplished by three methods.

The first method is to use the migration and purge functions to free up space. Migration and purge can be forced manually to run; see Section 8.4.1 *Forcing Migration* and Section 8.4.2 *Forcing Purge*. For a long term solution, it may be desirable to tune the migration and purge policies to cause the automatic migration and purge operations to occur more frequently. This method of increasing storage space works for all levels in the Class of Service hierarchy except the lowest level.

The second method is to add additional storage volumes to the storage class using the normal import and create resources procedures. See Section 8.1 *Adding Storage Space* on page 223.

The third method is to increase free tape storage space by repacking sparse volumes and reclaiming empty volumes. The **repack** program moves information from sparsely filled volumes to other volumes in the same storage class. When the **repack** is finished, a number of volumes will be cleared of data. To make these volumes ready for reuse, they must be processed by the **reclaim** program, which refreshes the

Core Server metadata that describes the volumes. See Section 8.4.3, *Repacking and Reclaiming Volumes*.

8.4.1. Forcing Migration

The Migration Purge Server runs migration periodically in the time interval specified in the migration policy. However, between these automatic migration runs, an administrator can use the *Active Storage Classes* window to force a migration to take place. When a migration run is forced, the run timer is reset. The next automatic migration will take place after the specified interval after the end of the forced run.

From the *Active Storage Classes* list window, select one or more migratable storage classes and then click on the Migration Controls pull-down button and select Start from the list. Note: This request will go straight to the Migration Purge Server without confirmation.

The Migration Purge Server stops the migration run when either the Migration Target threshold in the migration policy is reached or there are no more disk files or tape virtual volumes eligible for migration.

The only difference between a forced migration run and an automated migration run is that the forced run is initiated manually by an administrator. Therefore, a forced migration will not migrate files that do not meet the migration criteria.

8.4.2. Forcing Purge

The Migration Purge Server obtains the disk storage class statistics periodically. The period is set in the Migration Purge Server's specific configuration record. The Migration Purge Server then evaluates the used space and may start a purge run. Between these automatic purge runs, an administrator can use the *Active Storage Classes* window to force the start of a purge run for one or more storage classes.

From the *Active Storage Classes* window, select one or more purgeable storage classes, and then click on the Purge Controls pull-down button and select Start from the list. Note: This request will require confirmation before going to the Migration Purge Server.

The Migration Purge Server stops purging when the space used falls to the value specified in the purge policy or there are no more disk files eligible for purging.

The only difference between a forced purge run and an automated purge run is that the forced run is initiated manually by an administrator. Therefore, a forced purge will stop purging when the appropriate threshold is reached just as an automated purge run would.

8.4.3. Repacking and Reclaiming Volumes

Over time, the active data on a tape virtual volume may become sparse as files are deleted or migrated. The **repack** utility program provides the administrator with the ability to move data from sparse volumes to other volumes, allowing more efficient use of tape resources. The program can be invoked via the command line or from an SSM window. Running **repack** from the command line affords the greatest flexibility in its use and the opportunity to monitor its progress.

When **repack** processes tape volumes that contain file aggregates, it will move the aggregates to a new tape volume, leaving out those parts of the aggregates that are no longer pointed to by files.

Repack can also move disk data from one volume to another. **Repack** is frequently used to empty a failing disk volume by first setting it to RW (or perhaps RO depending on the nature of the failure), then moving its contents to other volumes.

Repack selects tape volumes in one of two ways. The administrator can provide a list of tape volumes to repack, or **repack** can select volumes based on a number of selection criteria. If **repack** is provided with a list of tape volumes to process, those volumes must be in RW, RO, EOM or EMPTY Condition. Volumes in RWC or DOWN Condition cannot be selected. Repacking RW and EMPTY tape volumes is permitted because the administrator has explicitly selected the tape. Repacking EMPTY tape volumes is also permitted because it is harmless to do so and it provides a way to prove that the volume is empty.

When **repack** selects tape volumes to process, it uses *number*, *storage class*, *family*, *retired*, *checked out*, and *percentage full* criteria to select a set of tape volumes. The selected volumes will be in RO or EOM condition. Volumes in RWC, EMPTY or DOWN Condition are never selected. Tapes that are in RW Condition and are also Retired will be treated as if they are in RO Condition and may be selected if they qualify under the other selection criteria. If the *-M* option is used, only retired tapes that meet the rest of the selection criteria will be selected.

Disk volumes to be repacked must be in RW, RO or EMPTY Condition. EMPTY volumes may be repacked because it is harmless to do so, the administrator has explicitly selected the disks, and it provides a way to prove that the volume is empty.

Repacking disks cannot be done via the SSM window; the program must be executed from a command line following the instructions given in the **repack** man page.

When the **repack** process completes successfully, the Number of Active Segments in the volume's display will be zero and the VV Condition will be EMPTY.

Note that tape volumes in EOM Condition go to EMPTY Condition when **repack** moves all of the data on the volumes to other volumes. Volumes in RW and RO Condition do not go to EMPTY Condition. Manually setting the Condition of an empty tape volume to EOM causes it to go to EOM Condition, then immediately to EMPTY Condition.

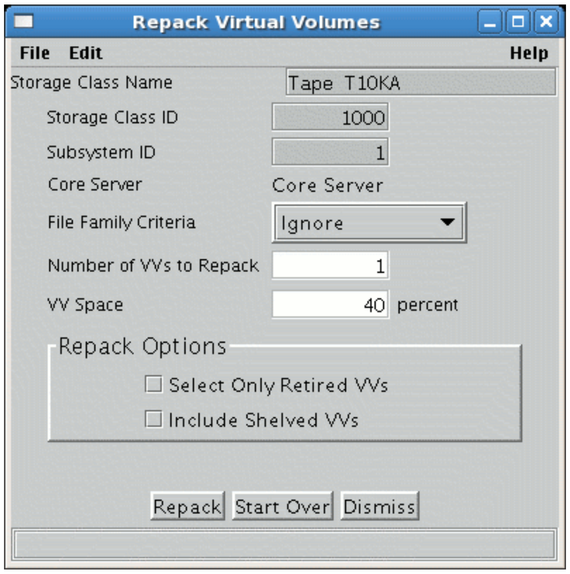
After a tape volume is successfully repacked, it remains in EMPTY state and is not available for storing data. To make the volume available, it must be reclaimed. Disk volumes can be placed back in service by changing the VV Condition to RWC.

Note that both disk and tape **repack** operations can put significant demands on system resources, particularly when repacking striped volumes. Since repack performs a tape to tape copy operation, it will consume twice as many tape drives as the stripe width of the volumes being repacked.

The **reclaim** program deletes empty tape virtual volumes and re-creates them with the same configuration as the original volume. The PV Name(s), Stripe Width, PV and VV Block Sizes, PV and VV Estimated Size and other parameters of the old VV are used to create a new VV with a new Virtual Volume ID (VVID). The metadata describing the old VV is deleted. HPSS requires that tapes be reclaimed, rather than reused, so that it can be certain that no references to the old VV exist or could be resolved.

Reclaimed tape volumes are entered into HPSS metadata in RWC Condition and are immediately available for use.

8.4.3.1. Repack Virtual Volumes Window


Repack Virtual Volumes

File Edit Help

Storage Class Name: Tape T10KA
Storage Class ID: 1000
Subsystem ID: 1
Core Server: Core Server
File Family Criteria: Ignore
Number of VVs to Repack: 1
VV Space: 40 percent

Repack Options
☐ Select Only Retired VVs
☐ Include Shelved VVs

This window provides an interface for repacking tape virtual volumes.

Field Descriptions

Storage Class Name. The name of the storage class that will be repacked.

Storage Class ID. The ID of the storage class that will be repacked.

Subsystem ID. The ID of the subsystem which contains the volumes to be repacked.

Core Server. The name of the Core Server that manages the volumes to be repacked.

File Family Criteria. Indication of whether to use File Family criteria. Click on the button to drop-down a list of available options. Select **Ignore** to indicate not to use the File Family criteria when selecting volumes for repack. Select the name of a file family to repack if you want to use File Family criteria. When a file family is selected, volumes assigned to that family will become candidates for repack. Volumes not in the selected family will not be candidates. Storage segments on repacked volumes will be transferred to other volumes assigned to the same family. If no writable tape volumes are available in the family, a new blank volume will be chosen and assigned to the family. If **Not in a family** is selected, volumes not assigned to any file family will become candidates for repack. Volumes that have been assigned to a file family will not be candidates.

Number of VVs to Repack. The maximum number of tape volumes to repack. If **repack** does not find this many candidate volumes, those that have been selected will be repacked and no error will result.

VV Space. This is a repack selection criterion expressed as a percentage. If the amount of occupied space on a candidate tape is less than this value, the tape will be considered for repacking. Tape volumes with larger amounts of occupied space will not be considered.

The occupied space percentage is calculated by summing the lengths of the files on the candidate tape and comparing that sum to the VV Estimated Size field in the storage map. If this ratio is less than the VV Space percentage, the volume is selected. Note that vacated files in file aggregates on tapes will not be counted as occupied space. A tape containing a relatively small number of large file aggregates from which a large percentage of the files have been deleted, may be selected by **repack**.

If VV Space is 100, the comparison is not performed and selection of tape volumes to repack is made using the remaining criteria.

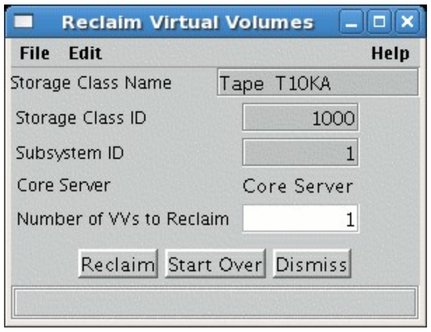
Repack Options.

- **Select Only Retired VVs**. If selected, **repack** selects only retired volumes in the indicated storage class. Retired volumes may be in RO, EMPTY or EOM condition. DOWN volumes are never selected. If this option is not selected, both retired and non-retired tape volumes in RO, EMPTY or EOM condition may be selected.
- **Include Shelved VVs**. Tape volumes which have been removed from HPSS using the shelf tape utility are eligible for repack.
-

Buttons

Repack. Press this button to start the **repack** program on the indicated storage class. Tape volumes that meet the repack criteria will be repacked. Status messages are displayed on the status bar at the start and end of the repack.

8.4.3.2. Reclaim Virtual Volumes Window



Storage Class Name: Tape T10KA
Storage Class ID: 1000
Subsystem ID: 1
Core Server: Core Server
Number of VVs to Reclaim: 1

This window is used to run the **reclaim** utility program which removes and recreates tape virtual volumes.

Field Descriptions

Storage Class Name. The name of the storage class selected for reclaiming.

Storage Class ID. The ID of the storage class selected for reclaiming.

Subsystem ID. The ID of the subsystem in which the volumes to be reclaimed reside.

Core Server. The name of the Core Server that will perform the reclaim operations.

Number of VVs to Reclaim. The number of tape virtual volumes in the storage class that will be reclaimed. If the number of candidate volumes is less than this value, all eligible volumes will be reclaimed. Retired volumes are never reclaimed.

Buttons

Reclaim. Press this button to start the **reclaim** utility program on the indicated storage class. Tape volumes that are described as EMPTY, and are not retired, will be reclaimed. Status messages are displayed on the status bar at the bottom of the window at the start and end of the reclaim.

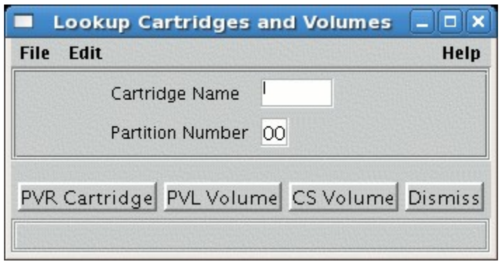
SSM invokes the **reclaim** utility program and passes it the storage class and number of tape volumes to be reclaimed, as entered on the *Reclaim Virtual Volumes* window. The utility program selects empty tape volumes from the storage class until it selects the required number or the supply of empty volumes is exhausted. The volumes are reclaimed and immediately become available for use.

The **reclaim** utility can also be invoked as a command line utility. Used this way, the user has more flexibility in choosing options and can monitor the program's progress. Refer to the **reclaim** man page for more information on how to invoke the utility.

8.5. Volume Management

This section describes how to monitor and manage PVL volumes, PVR cartridges, and Core Server Disk and Tape volumes.

8.5.1. Lookup Cartridges & Volumes Window



This window allows you to access information windows for PVR cartridges, PVL volumes, and Core Server volumes. After pressing a button to perform a lookup operation, this window will remain open to allow additional lookups.

Field Descriptions

Cartridge Name. The six-character cartridge label. A string exactly six characters long must be entered in this field.

Partition Number. This field is no longer used and should always be “00”.

Buttons

PVR Cartridge (tape only). Once you have filled in the cartridge name, clicking on this button will open the *PVR Cartridge Information* window for the specified cartridge. This metadata is created when the cartridge is successfully imported.

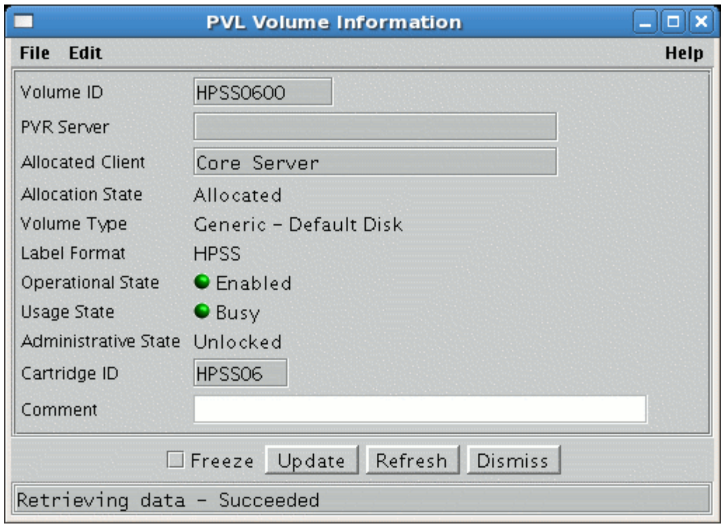
PVL Volume. Once you have filled in both fields, clicking on this button will open the *PVL Volume Information* window for the specified volume. This metadata is created when the volume is successfully imported.

CS Volume. Once you have filled in both fields, clicking on this button will open the *Core Server Disk Volume* or *Core Server Tape Volume* window for the specified volume. This metadata is created when the disk/tape storage resources are successfully created.

8.5.2. PVL Volume Information Window

The *PVL Volume Information* window allows the SSM user to view the data for imported volumes.

Before using the window, the user should know the 6-character label of the PVL volume. The **dumppv_pvl** utility can be used to list all the volumes being managed by the PVL and help determine the label.



Volume ID	HPSS0600
PVR Server	
Allocated Client	Core Server
Allocation State	Allocated
Volume Type	Generic - Default Disk
Label Format	HPSS
Operational State	● Enabled
Usage State	● Busy
Administrative State	Unlocked
Cartridge ID	HPSS06
Comment	

Retrieving data - Succeeded

This window allows you to view the information associated with a PVL volume.

Field Descriptions

Volume ID. The eight-character volume label for the volume being viewed.

PVR Server. The descriptive name of the PVR that owns the cartridge corresponding to the volume being viewed. This field is applicable only to tape volumes and is always blank for disk volumes.

Allocated Client. The descriptive name of the Core Server which manages the volume. This is the name of the Core Server specified when storage resources were allocated on the volume using the Create Disk Resources or Create Tape Resources operation. If no storage resources have been created on the volume, this field will be blank.

Allocation State. The usability of the volume. This includes whether the volume import and label operation completed successfully, whether storage resources have been created on the volume, and whether the volume is currently stored in the robot or removed to an external shelf or vault.

Possible values are:

- Allocated. Storage resources have been created on the volume, it is allocated to a Core Server, and it is available to the system for I/O.

- Unallocated. The volume has been successfully imported into the HPSS system and labeled. However, no storage resources have been created on it and it has not been allocated to a Core Server. It is therefore not available to the system for I/O.
- Allocated - On Shelf. Storage resources have been created on the volume, it is assigned to a Core Server, and it is available to the system. However, it has been removed from the tape robot (to a vault, for example), so there will be a delay for any mount operation while the volume is returned to the robot.
- Unlabeled. An error occurred during the mount/label operation during import. A volume in this state is not usable by HPSS. It should be exported and re-imported.
- Allocated - No Shelf Info. The volume has been allocated to a Core Server and is available to the system for I/O, but no shelf information is available because the PVL can not communicate with the PVR.

Volume Type. The HPSS media type corresponding to this volume.

Label Format. The type of label on the volume. Possible values are:

- HPSS - The volume has an ANSI label (it starts with an 80-byte block beginning with the characters "VOL1"). The owner field of the label contains "HPSS".
- Foreign - The volume has an ANSI label, but the owner field is something other than "HPSS".
- None - The volume has no label. A volume in this state is not usable by HPSS. It should be exported and re-imported.
- Data - The volume is an uninitialized tape.
- NonANSI - The label is an 80 byte non-ANSI record.
- NSL - The volume has a NSL label format (80 byte record w/o EOF).
- Unknown - The label of the volume failed. Status of label unknown.

Operational State. This will always be Enabled.

Usage State. Busy if the volume is in use; Idle otherwise.

Administrative State. This will always be Unlocked.

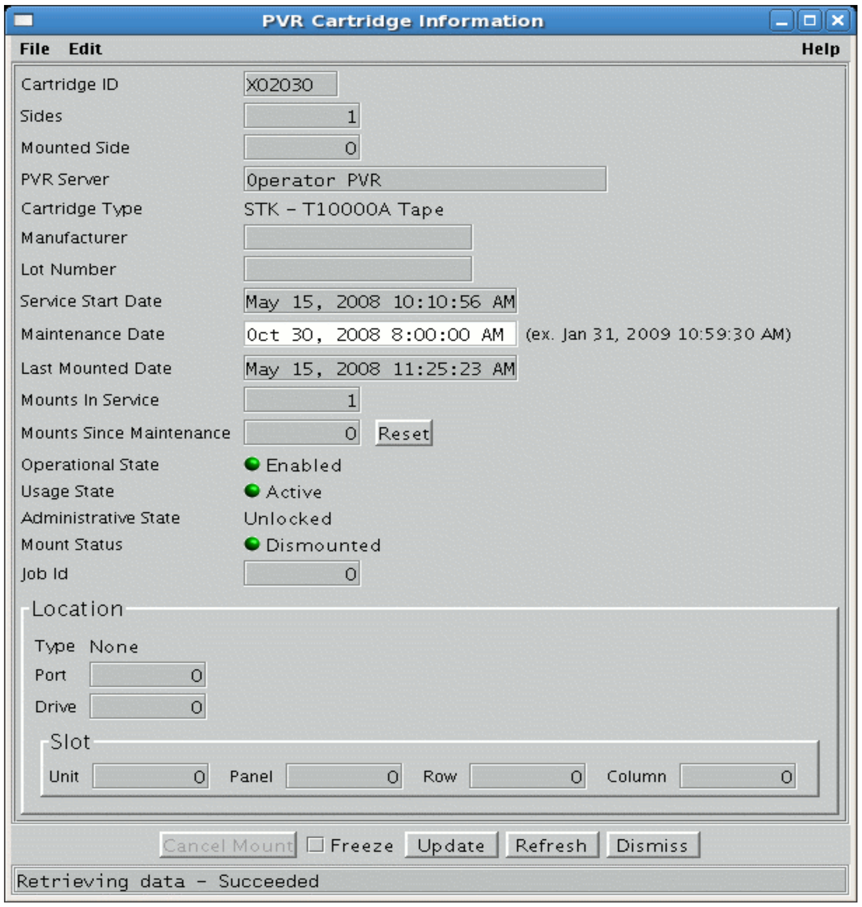
Cartridge ID. The six-character cartridge label for the volume.

Comment. This field provides a 128 character buffer in the PVL volume metadata which gives the administrator the opportunity to associate miscellaneous text with a volume. For example, a site may want to place a comment in this field that the volume (cartridge or disk) is out of service.

8.5.3. PVR Cartridge Information Window

The *PVR Cartridge Information* window allows the SSM user to view the data about a cartridge managed by one of the HPSS PVRs.

Before using the window, the user should know the 6-character label of the cartridge. The **dumppv_pvr** utility can be used to list all the cartridges being managed by the PVR and help determine the label.



Cartridge ID	X02030
Sides	1
Mounted Side	0
PVR Server	Operator PVR
Cartridge Type	STK - T10000A Tape
Manufacturer	
Lot Number	
Service Start Date	May 15, 2008 10:10:56 AM
Maintenance Date	Oct 30, 2008 8:00:00 AM (ex. Jan 31, 2009 10:59:30 AM)
Last Mounted Date	May 15, 2008 11:25:23 AM
Mounts In Service	1
Mounts Since Maintenance	0 Reset
Operational State	● Enabled
Usage State	● Active
Administrative State	Unlocked
Mount Status	● Dismounted
Job Id	0

Location

Type	None
Port	0
Drive	0

Slot

Unit	0	Panel	0	Row	0	Column	0

Retrieving data - Succeeded

This window allows you to view and update the information associated with an HPSS tape cartridge.

Note that the **Location Type** fields are represented differently for certain types of robots, for which **Port**, **Drive** and **Slot** (**Unit**, **Panel**, **Row**, and **Column**) may each be displayed as 0. If it is necessary to locate a cartridge in one of these robots, the robot's operator interface must be used.

This window contains three bookkeeping fields: **Maintenance Date**, **Mounts Since Maintenance**, and **Mount Status**. If the some maintenance is performed on the cartridge, such as re-tensioning it, the **Mounts Since Maintenance** field can be reset. The **Maintenance Date** field can also be reset. Note that resetting one field does not affect the other. The third operation involves resetting the **Mount Status** from Mount Pending to Dismounted via the **Cancel Mount** button. This operation is only required for handling a specific error condition.

Any changes made to fields on this window are sent directly to the PVR, and are effective immediately.

Field Descriptions

Cartridge ID. The six-character cartridge label.

Sides. The number of partitions (sides) on the cartridge. Currently always set to one.

Mounted Side. The partition currently mounted. Set to zero if the cartridge is not currently mounted;

note that this can also mean that side 0 of the cartridge is mounted.

PVR Server. The descriptive name of the PVR which manages the cartridge.

- **Cartridge Type**. The HPSS media type corresponding to this cartridge. This controls which the type of drive in which the cartridge can be mounted.

Manufacturer. The Manufacturer string specified when the cartridge was imported.

Lot Number. The Lot Number string specified when the cartridge was imported.

Service Start Date. The date and time when the cartridge was imported.

Maintenance Date. If the maintenance is performed on the cartridge, for example by re-tensioning it, this field can be set to record the date of the maintenance. The format for entering the date is given to the right of the field. This format can be changed with the -D option to the hpssgui startup script.

Last Mounted Date. The date and time when the cartridge was last mounted.

Mounts In Service. The number of times the cartridge has been mounted by HPSS since it was imported.

Mounts Since Maintenance. The number of times the cartridge has been mounted since its last maintenance. Click the **Reset** button to the right of this field to reset the value to zero. This field is not automatically reset when Maintenance Date is changed.

Operational State. This will always be Enabled.

Usage State. This should always be Active.

Administrative State. This will always be Unlocked.

Mount Status. The mount status of the cartridge. Possible values are:

- Mount Pending
- Mounted
- Dismount Pending
- Dismounted
- Eject Pending
- Checked In
- Check-In Pending
- On Shelf
- Check-Out Pending
- Move Pending
- Move Shelf Pending

Job Id. The job identifier associated with the cartridge. A value of zero indicates that there is not job associated with the cartridge (i.e. when the cartridge is not mounted).

Location Type. The type of cartridge location information being displayed. Possible values are:

- None - No location information is available.

- Port - The location is a port number. [This option is currently not used.]
- Drive - The location is a drive ID number.
- Slot - The location is a slot specification.

The following fields are filled in (non-zero) based on the **Location Type** and whether or not the PVR has the information:

> **Port**. The port number where the cartridge is located. This option is currently not used and thus will always be zero.
>
> **Drive ID**. The drive ID number where the cartridge is located. This field is valid when the **Location Type** is Drive. It is used/valid when the cartridge is Mounted; it is not used/valid (and thus will have the value of zero) when the cartridge is not Mounted. The SCSI PVR also uses this field to support the **Enforce Home Location** feature.
>
> **Slot**. The slot address (**Unit**, **Panel**, **Row** and **Column**) for the cartridge. These fields are valid only if **Location Type** is Slot and the PVR supports this (STK and AML). The fields will be zero when it's not valid.

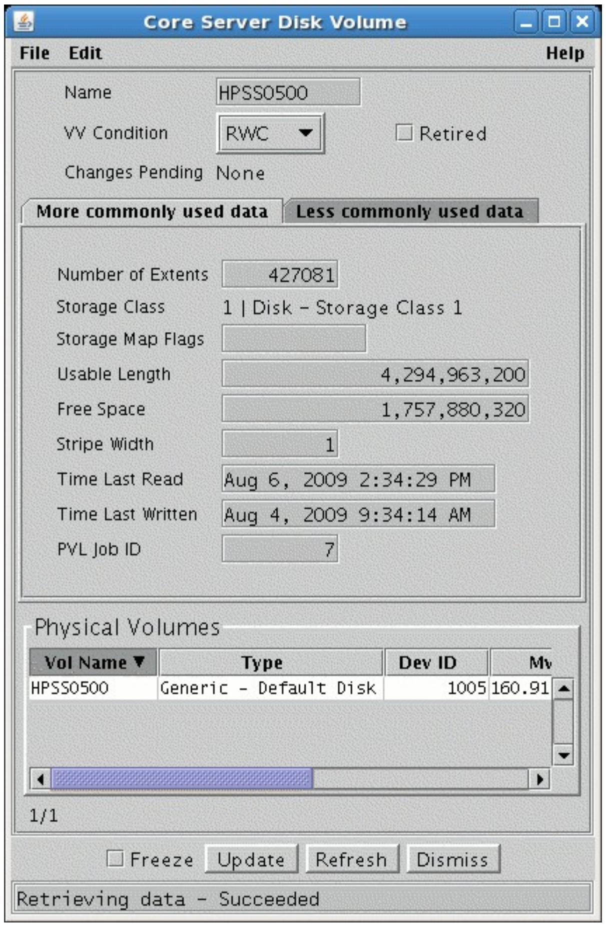
Buttons

Cancel Mount. When Mount Status is displaying Mount Pending, the **Cancel Mount** button to the right of the field will be sensitized. If you click the button, a large warning will be displayed, and you will have to confirm your request by clicking **Cancel Mount** again. This will send a request to the PVR to cancel the mount request for the cartridge. As noted in the on-screen warning, this should only be done as a last-ditch attempt to cancel a mount request which cannot be canceled in any other way.

⚠ *Note that this will not dismount a mounted tape.*

8.5.4. Core Server Volume and Segment Windows

8.5.4.1. Core Server Disk Volume Information Window


Core Server Disk Volume

File Edit Help

Name	HPSS0500
VV Condition	RWC
Retired	☐
Changes Pending	None

More commonly used data | Less commonly used data

Number of Extents	427081
Storage Class	1 | Disk - Storage Class 1
Storage Map Flags	
Usable Length	4,294,963,200
Free Space	1,757,880,320
Stripe Width	1
Time Last Read	Aug 6, 2009 2:34:29 PM
Time Last Written	Aug 4, 2009 9:34:14 AM
PVL Job ID	7

Physical Volumes

Vol Name	Type	Dev ID	Mv
HPSS0500	Generic - Default Disk	1005	160.91

1/1

☐ Freeze Update Refresh Dismiss

Retrieving data - Succeeded

This window displays information about a disk volume as represented by the Core Server.

Field Descriptions

Name. The ASCII name of the first physical volume that is a part of the disk virtual volume. The entire virtual volume can be referred to by this name.

VV Condition. This is the administrative control for the disk virtual volume. It will have one of five values: RWC, RW, RO, EMPTY or DOWN.

- In RWC condition, the volume can be read and written. This is the normal operational state.
- In RW condition, the volume can be read and written, but new disk storage segments may not be created on the volume.
- In RO condition, the volume can be read but not written. New storage segments cannot be created on the volume.
- In EMPTY condition, the volume cannot be read or written. All storage segments have been removed from the volume, and new segments cannot be created. Since this condition represents a physical state of the volume, the operator may not change the volume to this condition. EMPTY is displayed by the server when the Condition is RO and there are no storage segments in the

volume.

- In DOWN condition, the volume cannot be read, written or mounted. This condition can be used to make a disk unavailable to the system.

Change the **VV Condition** of a disk virtual volume by selecting the desired condition from the drop down menu and then pressing the Update button.

Changes Pending. If there are any VV Condition changes for this volume pending in the Core server, **Changes Pending** will be "Yes" with a red bullet. Otherwise, **Changes Pending** will be None.

Retired. When checked, the volume is retired. New storage segments will not be created on the volume.

More Commonly Used Data Tab

Number of Extents. The number of disk storage segment extents allocated in this virtual volume.

Storage Class. The storage class to which the disk virtual volume is assigned.

Storage Map Flags. Normally this field is blank, but if "Offline" is displayed in this field, the disk volume is not available for use because the PVL mount job failed. The Core Server received an error when it tried to mount one or more of the disks in the volume the last time it started. Accesses to the disk will receive an error. New storage segments will not be created on the disk. The volume can be brought back on-line only by correcting the mount problem and restarting the Core Server. If more than one of the underlying physical volumes is off-line, the virtual volume will remain off-line until all physical volumes are on-line

Usable Length. The amount of disk space that can be used for storing storage segments. This is the Actual Length of the disk volume minus certain amounts set aside for system use.

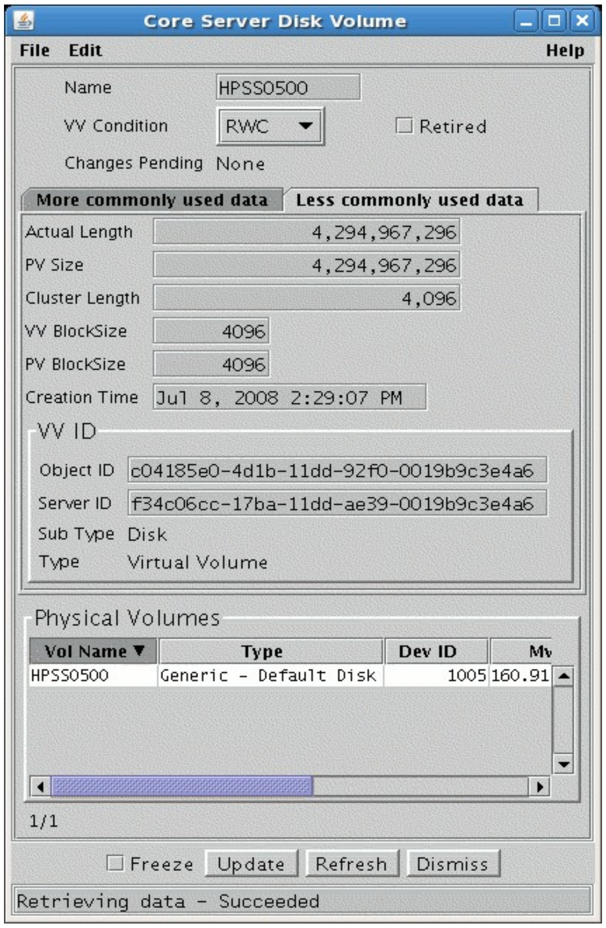
Free Space. The number of bytes on the disk virtual volume that are not assigned to storage segments. Free Space is initially set to the value of Usable Length, then decremented and incremented by the size of disk storage segments as they are created and deleted. Since disk storage segments are allocated in whole Clusters, they are usually larger than the amount of storage needed to store the segment of the file. This creates “slack space” which is space allocated to files, but not filled with file data. The amount of disk space in use for any volume, which is the Usable Length minus the Free Space, will always be somewhat greater than the sum of the lengths of the file segments stored on the volume. Since unallocated disk space may become fragmented, it may not be possible to create new disk storage segments on a volume even if the amount of Free Space appears to be large. The size of the largest free extent on the volume is logged by the Core Server when this happens.

Stripe Width. The number of physical volumes in the stripe group that makes up the virtual volume. All volumes in HPSS are considered to be striped, even if this value is one.

Time Last Read. The date and time the volume was last read. This value is not stored in metadata, but is initialized to zero when the Core Server starts and then updated each time the disk is read. This field is not affected by transaction aborts.

Time Last Written. The date and time the volume was last written. This value is not stored in metadata, but is initialized to zero when the Core Server starts and then updated each time the disk is written. This field is not affected by transaction aborts.

PVL Job ID. The PVL job ID in which this volume is mounted. This field will be zero if the volume is not mounted.


Core Server Disk Volume

Name	HPSS0500
VV Condition	RWC
Retired	☐
Changes Pending	None

More commonly used data | Less commonly used data

Actual Length	4,294,967,296
PV Size	4,294,967,296
Cluster Length	4,096
VV BlockSize	4096
PV BlockSize	4096
Creation Time	Jul 8, 2008 2:29:07 PM

VV ID

Object ID	c04185e0-4d1b-11dd-92f0-0019b9c3e4a6
Server ID	f34c06cc-17ba-11dd-ae39-0019b9c3e4a6
Sub Type	Disk
Type	Virtual Volume

Physical Volumes

Vol Name ▼	Type	Dev ID	Mv
HPSS0500	Generic - Default Disk	1005	160.91

1/1

☐ Freeze Update Refresh Dismiss

Retrieving data - Succeeded

Less Commonly Used Data Tab

Actual Length. The length of disk virtual volume in bytes. This length includes all of the space set aside for system use. See **Usable Length**.

PV Size. The length in bytes of a physical volume in this disk virtual volume. All physical volumes in a disk virtual volume must be the same length.

Cluster Length. The size of the allocation unit, in bytes. When disk storage segments are created on the volume, they are created at a length that will be a multiple of this value.

VV Block Size. The virtual volume block size. This is the number of bytes written from a data stream to an element of the striped volume before the stream switches to the next element of the stripe.

PV Block Size. The size, in bytes, of the media data block.

Creation Time. The date and time the disk volume was created in the Core Server.

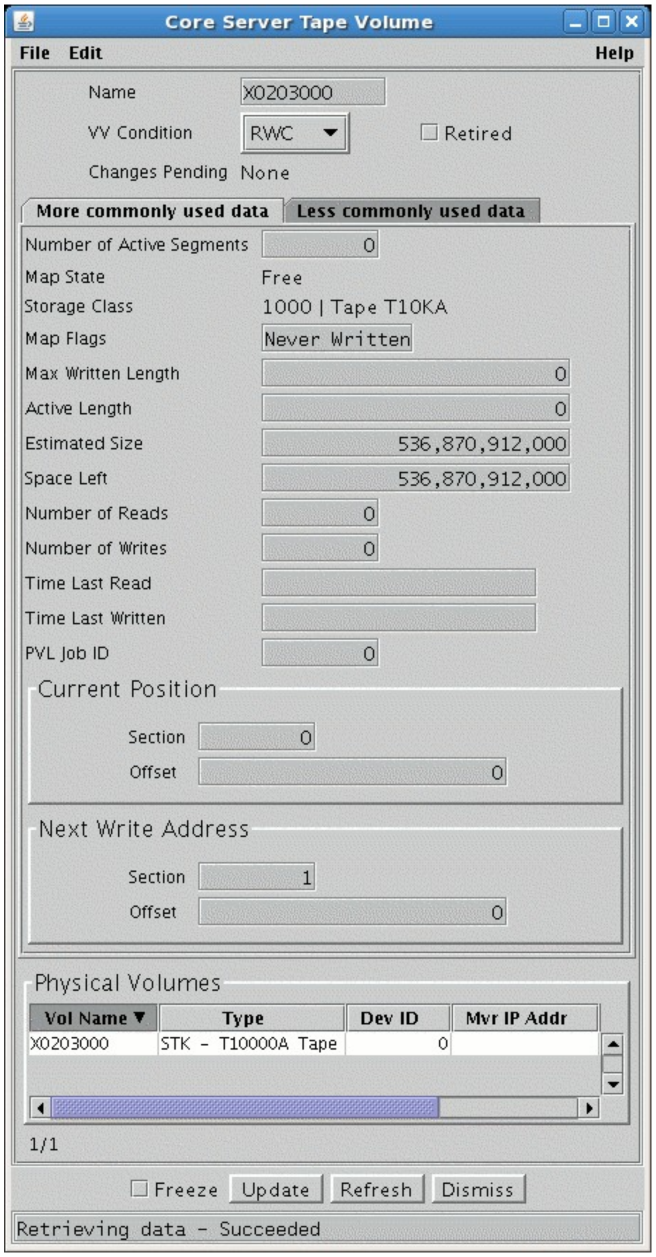
VV ID. The detailed description of the disk virtual volume ID associated with this disk virtual volume.

Physical Volumes

This is a table of physical volume attributes for the physical volumes that make up this disk virtual volume.

> **Vol Name.** The ASCII name of the physical volume.
>
> **Type.** The media type.
>
> **Dev ID.** The ID of the device the physical volume is mounted on.
>
> **Mvr IP Addr.** The IP address of the Mover that operates this physical volume.
>
> **Mvr.** The descriptive name of the Mover that operates this physical volume.
>
> **Mvr Host.** The name of the host on which the Mover runs.

8.5.4.2. Core Server Tape Volume Information Window



Name: X0203000
VV Condition: RWC
☐ Retired
Changes Pending: None

Number of Active Segments	0
Map State	Free
Storage Class	1000 | Tape T10KA
Map Flags	Never Written
Max Written Length	0
Active Length	0
Estimated Size	536,870,912,000
Space Left	536,870,912,000
Number of Reads	0
Number of Writes	0
Time Last Read	
Time Last Written	
PVL Job ID	0

Current Position
Section: 0
Offset: 0

Next Write Address
Section: 1
Offset: 0

Physical Volumes

Vol Name ▼	Type	Dev ID	Mvr IP Addr
X0203000	STK - T10000A Tape	0	

1/1

Retrieving data - Succeeded

This window displays information about a tape volume as represented by the Core Server.

Field Descriptions

Name. The ASCII name of the first physical volume that is a part of the tape virtual volume. The entire virtual volume can be referred to by this name.

VV Condition. This is the administrative control for the tape virtual volume. It will have one of six values: RWC, RW, RO, EOM, EMPTY or DOWN.

- In RWC condition, the volume can be read and written. This is the normal operational state.
- In RW condition, the volume can be read and written, but new tape storage segments may not be created on the volume.
- In RO condition, the volume can be read but not written. New storage segments cannot be created on the volume.
- In EOM condition, the volume can be read but not written. One or more of the tapes has been written to its end and the tape virtual volume is now full. New storage segments cannot be created on the volume. The volume condition can be changed to DOWN, but not to any other condition.

Note that setting a volume to EOM is irreversible. Don't use this setting unless you're sure it's what you want.

- In EMPTY condition, the volume cannot be read or written. The volume has reached EOM, and all storage segments have been removed from the volume. New storage segments cannot be created. Volumes cannot be changed manually to EMPTY condition. EMPTY is reported by the Core Server when the volume has reached EOM and no storage segments remain on the volume.
- In DOWN condition, the volume cannot be read, written or mounted. This condition can be used to make a tape unavailable to the system.

Change the condition of a tape virtual volume by selecting the desired condition from the drop down menu and then pressing the Update button.

Changes Pending. If there are any VV Condition changes for this volume pending in the Core server, **Changes Pending** will be "Yes" with a red bullet. Otherwise, **Changes Pending** will be None.

Retired. When checked, the volume is retired. New storage segments will not be created on the volume. Retired volumes may not be reclaimed.

More Commonly Used Data Tab

Number of Active Segments. The number of storage segments stored in this virtual volume.

Map State. This is the primary operational state flag of the tape virtual volume. It can take on one of the following values:

- Free - The volume is available to be written.
- Allocated - The volume is assigned to a tape write operation and is being written.
- EOM - The volume has reached End Of Media and can no longer be written.
- Empty - The volume has reached End Of Media and all storage segments have been deleted.
- Deny - The volume is not eligible to have a new tape storage segment created on it. Tape read operations are permitted, and the last storage segment on the tape may be written under certain circumstances.

The **Map State** field is displayed to provide an additional level of detailed information about the tape volume. It cannot be changed manually. Use the **VV Condition** to change the operational condition of the volume.

A tape volume starts in Free state, which indicates that it can be selected to receive a storage segment and its data. When data is being written to a volume, it will be shown in Allocated state. When the End Of Media marker is reached during a tape write operation, the volume will enter EOM state. If the number of segments on the volume drops to zero after reaching EOM, the state will change to Empty. The volume may be placed in Deny state by the server depending on the setting of the VV Condition.

Storage Class. The storage class to which the tape virtual volume is assigned.

Map Flags. Information about the state of the volume. Contains one of the following values:

•Never Written – Data has not been written to the volume.

•Retired – New data will not be written to this volume.

•Repack Output – All files on this volume were written there by the repack process.

Max Written Length. The total number of bytes that have been written to the volume. Each time the volume is written, this number is incremented by the number of bytes written. This number is not decremented when storage segments are deleted from the volume.

Active Length. The sum of the number of bytes in active storage segments. As segments are added to the volume, this number increases. As segments are deleted, this number decreases.

Estimated Size. The estimated size of the volume in bytes. It is only an estimate of the number of bytes that can be written on a tape volume. Depending on the compression characteristics of the data, the actual number of bytes may be smaller or greater.

Space Left. The estimated number of bytes that may still be written to the tape virtual volume. It is initially set to **Estimated Size** and is decremented as storage segments are written. It is not incremented when storage segments are deleted.

Because of the variable compressibility of data written to tapes, this value may behave strangely. For instance, the end of the media (EOM) may be reached earlier than estimated, causing this value to become 0 quite suddenly. Conversely, EOM may be reached later than expected. In this case, Space Left will be decremented until it reaches a value of 1, then remain at 1 until EOM is reached.

Number of Reads. The number of times the tape volume has been read.

Number of Writes. The number of times the tape volume has been written.

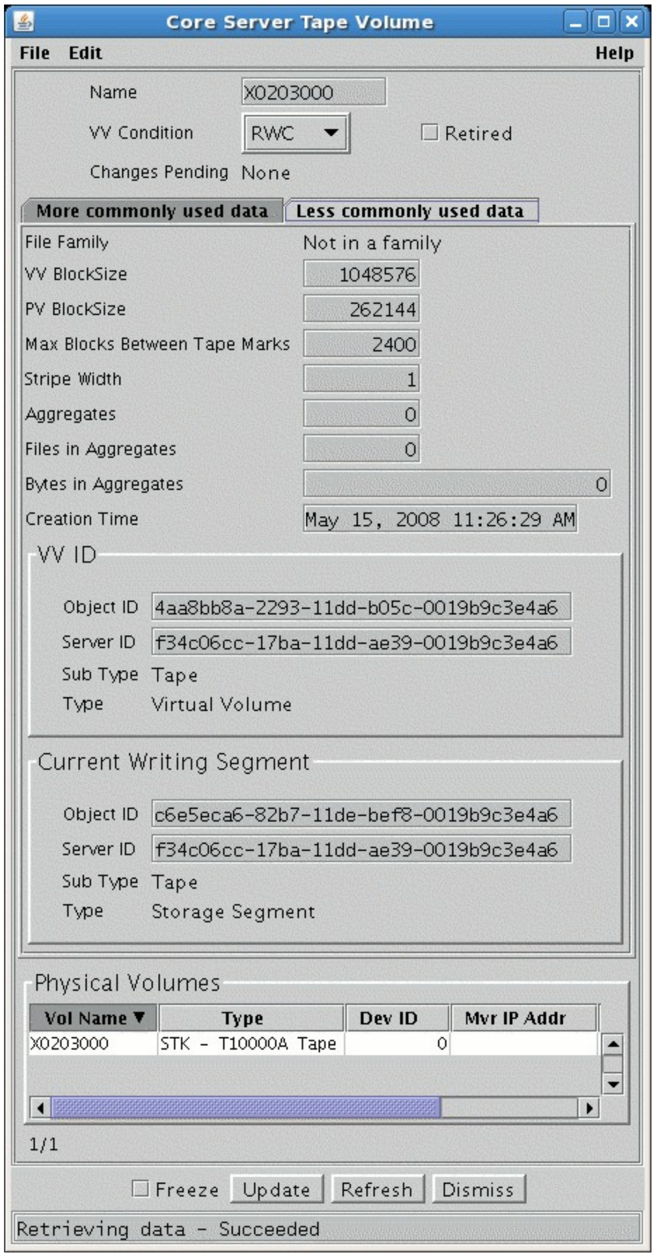
Time Last Read. The date and time the volume was last read. If this field is blank, the tape volume has not been read.

Time Last Written. The date and time the volume was last written. If this field is blank, the tape volume has not been written.

PVL Job ID. The PVL job ID in which this volume is mounted. This field will be zero if the volume is not mounted.

Current Position. The address to which the tape is positioned, expressed as an HPSS Relative Stripe Address. The Relative Address consists of two parts, the tape section and the byte offset. Tape sections are the data written between a pair of tape marks. The byte offset is the offset from the beginning of the current section to the place where the tape read or write is to take place. When reading the tape, this address is set to the initial location where the tape read begins, then advances as the read progresses. When writing a tape, this value is set to the **Next Write Address**, and then advances as the tape is written. Updates to this field occur when I/O operations complete, so in some cases significant amounts of time may pass between updates.

Next Write Address. The next address that will be written on the tape volume, expressed as an HPSS Relative Stripe Address.



Name X0203000

VV Condition RWC ☐ Retired

Changes Pending None

File Family	Not in a family
VV BlockSize	1048576
PV BlockSize	262144
Max Blocks Between Tape Marks	2400
Stripe Width	1
Aggregates	0
Files in Aggregates	0
Bytes in Aggregates	0
Creation Time	May 15, 2008 11:26:29 AM

VV ID

Object ID	4aa8bb8a-2293-11dd-b05c-0019b9c3e4a6
Server ID	f34c06cc-17ba-11dd-ae39-0019b9c3e4a6
Sub Type	Tape
Type	Virtual Volume

Current Writing Segment

Object ID	c6e5eca6-82b7-11de-bef8-0019b9c3e4a6
Server ID	f34c06cc-17ba-11dd-ae39-0019b9c3e4a6
Sub Type	Tape
Type	Storage Segment

Physical Volumes

Vol Name ▼	Type	Dev ID	Mvr IP Addr
X0203000	STK - T10000A Tape	0	

1/1

Retrieving data - Succeeded

Less Commonly Used Data Tab

File Family. The family to which the volume is assigned, if any. If it is not assigned to a family, it is assigned to the default family, family zero.

VV Block Size. The virtual volume block size. This is the number of bytes written from a data stream to an element of the striped volume before the stream switches to the next element of the stripe.

PV Block Size. The size, in bytes, of the media data block.

Max Blocks Between Tape Marks. The maximum number of media blocks that will be written to this tape before a tape mark is written. The product of this value and the Media Block Size defines the Physical Volume Section Length.

Stripe Width. The number of physical volumes in the stripe group that makes up the virtual volume. All volumes in HPSS are considered to be striped, even if this value is one.

Aggregates. The number of tape aggregates on this volume.

Files in Aggregates. The number of files in the tape aggregates on this volume.

Bytes in Aggregates. The number of bytes stored in the tape aggregates on this volume.

Creation Time. The date and time the tape volume was created in the Core Server.

VV ID. The detailed description of the tape virtual volume ID associated with this tape virtual volume. See Section Error: Reference source not found *Error: Reference source not found* on page Error: Reference source not found for a description of the subfields of this field.

Current Writing Segment. The detailed description of the tape storage segment ID associated with the segment that is found at the end of the tape volume. For various reasons, some tape volumes do not have a value in this field. It will be displayed filled with blanks in that case.

Physical Volumes

This is a table of physical volume attributes for the physical volumes that make up this tape virtual volume.

> **Vol Name.** The ASCII name of the physical volume.
>
> **Type.** The media type.
>
> **Dev ID.** The ID of the device the physical volume is mounted on. If the volume is not mounted, this field will be zero.
>
> **Mvr IP Addr.** The IP address of the Mover that operates this physical volume. If the volume is not mounted, this field will be zeros.
>
> **Mvr.** The descriptive name of the Mover that operates this physical volume. If the volume is not mounted, this field will be blank.
>
> **Mvr Host.** The name of the host on which the Mover runs. If the volume is not mounted, this field will be blank.

8.5.5. Changing Core Server Volume Condition

HPSS provides a single control that sets the operational condition of a disk or tape virtual volume. The control is found on the *Core Server Tape Volume* and *Core Server Disk Volume* windows and is called the VV Condition.

VV Condition controls the availability of the volume for the following actions:

- Creation of new storage segments
- Reading of existing storage segments
- Writing of existing storage segments
- Mounting of tape media

Tape volumes have six possible settings for VV Condition:

- **RWC** - Read, Write, Create
- **RW** - Read/Write
- **RO** - Read Only
- **EOM** - End of Media reached
- **EMPTY** – The volume reached EOM, and all data has been removed.
- **DOWN** - Volume not available

Disk volumes have five possible settings for VV Condition:

- **RWC** - Read, Write, Create
- **RW** - Read, Write
- **RO** - Read Only
- **EMPTY** – The volume is in RO condition and all data has been removed.
- **DOWN** - Volume not available

A volume in RWC condition is fully enabled for use. This is the normal setting for all disk volumes and for tape volumes that have not reached EOM. Storage segments can be created on the volume; existing segments can be read and written. Volumes in RWC condition can be changed to any other condition.

Existing storage segments on disk volumes in RW condition can be read and written, but new segments cannot be created. This setting is useful for emptying a disk through attrition. Disk volumes can be emptied more quickly by setting the Condition to RW, then purging or repacking the disk. Volumes in RW condition can be changed to any other condition except as limited by the Retire flag (see below).

RO condition is similar to RW condition, but writes to existing disk storage segments are not permitted.

Storage segments on tape volumes in RO condition can be read, but not written. New segments cannot be created. Tapes in RO condition are similar to tapes in EOM condition with the exception that their condition can be changed to RWC or RW while tapes in EOM condition cannot. Volumes in RO condition can be changed to any other condition except as limited by the Retire flag (see below).

RW condition is similar to RO condition, but allows the last segment of a tape volume to be written as long as it remains writable. Note that only the last storage segment on a tape volume may be writable, while all the others are never writable.

Tape volumes enter EOM condition when the end of media marker is reached while writing the tape or when the administrator sets the Condition to EOM. EOM condition is similar to RO condition - segments can be read, but not written. Unlike RO condition, tapes in EOM condition can only be changed to DOWN. EOM volumes cannot enter either RWC or RO condition.

Volumes in DOWN condition cannot be read, written, created on or mounted. This setting effectively removes the volume from the system while maintaining the records of the storage segments on it and is useful for dealing with failed disk or tape volumes. Disks in DOWN condition can be changed to RWC, RW or RO condition. Tapes in DOWN condition can be changed to RWC, RW or RO condition if they have not reached EOM. Tapes that have reached EOM can only be changed from DOWN to EOM.

Disk or tape volumes may be reported in EMPTY condition, but the administrator can never change a disk or tape condition to EMPTY manually. This condition will be reported when a tape in EOM condition becomes empty, or a disk in RO condition becomes empty.

The Retire flag controls entry into RWC condition. When the Retire flag is set, the Condition cannot be changed to RWC, but all other changes are permitted subject to the rules outlined above. By preventing entry to RWC condition, retired volumes can be read and written, but new storage segments cannot be added to the volume. Normal attrition will gradually empty the volume.

To retire a disk or tape volume, set the Condition to any setting other than RWC (typically RW for disks and RO for tapes) and then set the Retire flag. These two changes can be done in a single update of the *Core Server Disk Volume* or *Core Server Tape Volume* window. Note that tapes do not need to be in EOM condition to be retired.

The Retire flag can be cleared if necessary. Once cleared, the Condition can be changed to RWC, subject to the rules outlined above.

8.5.6. Moving PVR Cartridges to Another PVR

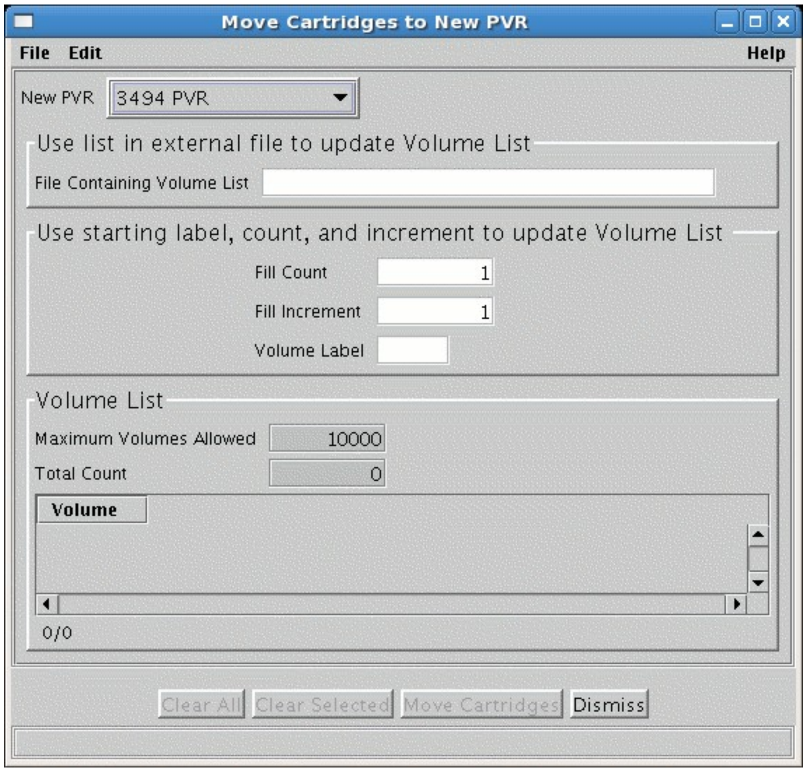
The intent of the Move Cartridge operation is to notify HPSS that a cartridge has been moved from one tape library to another.

Note that this operation will not cause a cartridge to move. Instead, the cartridge should be moved from the old library to the new library manually before informing HPSS. The operation will fail if the cartridge has not been ejected from its original robot and injected into the new robot. See the operator manual(s) for the specific robots involved to determine the procedures for manually ejecting and injecting a cartridge. Either the source or destination PVR (or both) during a move operation may be an operator mounted set of drives instead of a robot.

Specifically for shelf tapes, when a Move Cartridge is issued to a shelf tape, the target PVR will check whether or not the cartridge is physically in the target library. If it is not, then it will be treated as a move of a shelf tape to a shelf tape in the target library (i.e. maintain its checked-out / shelf status). If it is physically in the target library, then it will be treated as a move of a shelf tape to an online cartridge in the target library.

Use the *Move Cartridges to New PVR* window in the following section.

8.5.6.1. Move Cartridges Window



New PVR: 3494 PVR

Use list in external file to update Volume List

File Containing Volume List:

Use starting label, count, and increment to update Volume List

Fill Count	1
Fill Increment	1
Volume Label	

Volume List

Maximum Volumes Allowed	10000
Total Count	0

Volume

0/0

This window allows you to change which PVR owns a set of cartridges. Before initiating the request from the SSM window, the cartridges must already be physically placed into a tape library managed by the new PVR.

The list of the volumes to be moved may be constructed in any of three ways. Each volume name may be typed in one at a time, or a list of volume names may be automatically generated from a single entry, or a list of volume names may be specified from an input file. Volume names are not entered directly into the Volume List at the bottom of the window.

Any of the three entry methods may be repeated multiple times on the same window to add additional volumes to the list. All three entry methods or any combination of them may be used in succession on the same window.

To add a single volume name to the Volume List, set the **Fill Count** and the **Fill Increment** fields each to 1 and type the volume name into the **Volume Label** field. The volume name will be added to the Volume List.

To automatically generate a list of volume names in the Volume List, set the **Fill Count** to the desired number of volumes. Set the **Fill Increment** to the number by which each automatically generated label should differ from the previous one. Then type the starting volume name into the **Volume Label** field. The specified number of volume names will be added to the Volume List, each one larger than the previous entry by the specified **Fill Increment**.

Example:

```
        Fill Count = 6
```

```
Fill Increment = 10
Volume Label = "AA0070"

Labels automatically inserted into Volume List:
"AA0070"
"AA0080"
"AA0090"
"AA0100"
"AA0110"
"AA0120"
```

When an addition produces overflow in a column, numerical columns are carried over properly to alphabetic columns and vice versa.

Example:

```
Fill Count = 6
Fill Increment = 2000
Volume Label= "AA7329"

Labels automatically inserted into Volume List:
"AA7329"
"AA9329"
"AB1329"
“AB3329”
“AB5329”
“AB7329”
```

The filling will not occur and an error will be displayed if the specified values would generate an invalid volume label (e.g., one greater than zzz999).

To specify a list of volumes from a file, create a file containing the name of each volume to be moved on a separate line. Volume names must be six alphanumeric characters. No other characters are allowed on the line. The file must be accessible from the host on which the hpssgui is executing. Enter the name of the file in the **File Containing Volume List** field. The volume names from the file will be added to the Volume List.

Each cartridge in the list is sent to the System Manager as a separate move request. If an error occurs on a request, SSM stops processing the list of cartridges at that point. The window status line will report the number of successful moves, and the cartridge name that caused the failure. For example, if you try to move a cartridge to a PVR, and that cartridge is already controlled by that PVR, an error will occur. Therefore, if a multi-cartridge move fails, you cannot click the **Move Cartridge** button to retry the same list of cartridges, because the ones that succeeded before will now fail. You must first remove the successfully moved cartridges from the list. The **Clear Selected** button is provided for this purpose.

Field Descriptions

New PVR. The descriptive name of the PVR to which the cartridges will be moved.

File Containing Volume List. The name of an external file containing a list of cartridge labels to be added to the end of the Volume List.

Fill Count. The number of cartridge labels to be added to the end of the list when the **Volume Label** field is next modified. This number may be one or greater. If the list fills up before the **Fill Count** is exhausted, filling stops, and a message box is displayed (see Maximum Volumes Allowed below).

Fill Increment. In a multiple-cartridge fill, this field determines how each new cartridge label is generated from the previous one. A cartridge label is six alphanumeric characters. The **Fill Increment** is added to the least significant part of a cartridge label to generate a new label.

Volume Label. The cartridge label to be added to the end of the Volume List. If **Fill Count** is greater than 1, this will be the first label of a fill sequence. The label must be exactly six characters long.

Maximum Volumes Allowed. The maximum number of cartridge labels that will be allowed in the Volume List (10,000).

Total Count. The total number of labels currently in the Volume List.

Volume List. A list of cartridge/volume labels specifying the cartridges to be moved. You cannot enter labels directly into this list, but must construct it in one of the three ways described above. Use the scrollbar to move up and down the list of labels.

You can select one or more labels from the list for the purpose of removing them from the list (see **Clear Selected** below).

Buttons

Clear All. Clears the Volume List, and resets **Fill Count** and **Fill Increment** to 1.

Clear Selected. If one or more cartridges are selected (highlighted) in the Volume List, clicking on this button will remove them from the list. Note that this does not actually delete anything from HPSS.

Move Cartridges. Begins moving the listed cartridges. A start message is displayed on the message line at the bottom of the window, and all window features are disabled, except for the **Dismiss** button; this prevents you from modifying any data on the window while the move is in progress.

It is a good idea to have the *HPSS Alarms and Events* window open while moving proceeds, so that all relevant log messages can be viewed. Move operation initiation, completion and errors will be reported in the *HPSS Alarms and Events* window.

While moving proceeds, status messages are displayed on the status bar. When the move is finished, a completion message is displayed and the window features are reenabled. At this point new data can be entered for a new move.

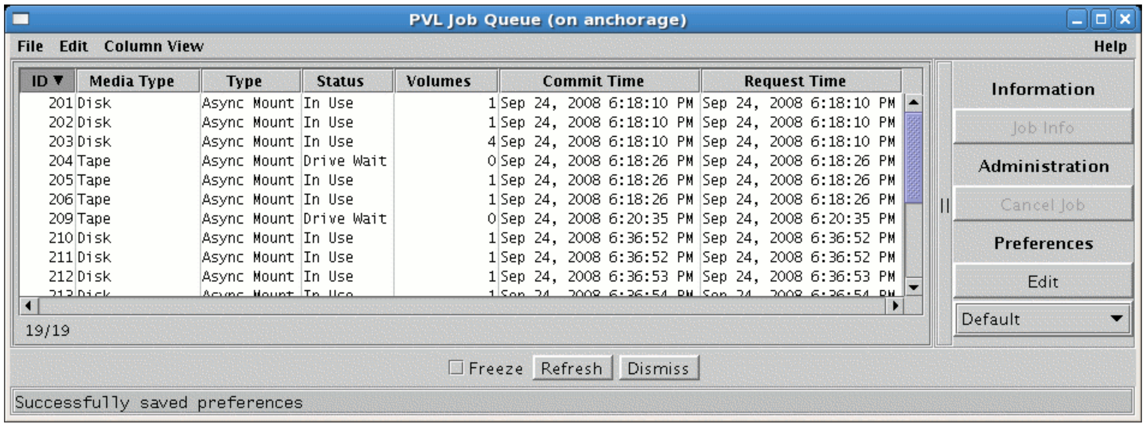
8.6. Monitoring and Managing Volume Mounts

Volume mounts are performed by the PVL. Every volume mount is associated with a PVL job. The *PVL Job Queue* window, lists all PVL jobs and the *PVL Request Information* window lists detailed information about a single PVL job.

The PVL reports every tape mount request in the *Tape Mount Requests* window. Tapes which need to be returned to the robot from a shelf or vault location are reported in the *Tape Check-In Requests* window.

Each of these windows is described in detail in this section.

8.6.1. PVL Job Queue Window



ID ▼	Media Type	Type	Status	Volumes	Commit Time	Request Time
201	Disk	Async Mount	In Use	1	Sep 24, 2008 6:18:10 PM	Sep 24, 2008 6:18:10 PM
202	Disk	Async Mount	In Use	1	Sep 24, 2008 6:18:10 PM	Sep 24, 2008 6:18:10 PM
203	Disk	Async Mount	In Use	4	Sep 24, 2008 6:18:10 PM	Sep 24, 2008 6:18:10 PM
204	Tape	Async Mount	Drive Wait	0	Sep 24, 2008 6:18:26 PM	Sep 24, 2008 6:18:26 PM
205	Tape	Async Mount	In Use	1	Sep 24, 2008 6:18:26 PM	Sep 24, 2008 6:18:26 PM
206	Tape	Async Mount	In Use	1	Sep 24, 2008 6:18:26 PM	Sep 24, 2008 6:18:26 PM
209	Tape	Async Mount	Drive Wait	0	Sep 24, 2008 6:20:35 PM	Sep 24, 2008 6:20:35 PM
210	Disk	Async Mount	In Use	1	Sep 24, 2008 6:36:52 PM	Sep 24, 2008 6:36:52 PM
211	Disk	Async Mount	In Use	1	Sep 24, 2008 6:36:52 PM	Sep 24, 2008 6:36:52 PM
212	Disk	Async Mount	In Use	1	Sep 24, 2008 6:36:53 PM	Sep 24, 2008 6:36:53 PM

19/19

Successfully saved preferences

This window shows all outstanding jobs in the PVL. From this window, the user can issue a request to view more information for a particular PVL job or to cancel it. Each PVL job represents a volume mount (or series of mounts for a striped disk or tape volume).

Field Descriptions

Job List. This is the main part of the window, consisting of a table of job information, a title line containing labels for each column, and vertical and horizontal scrollbars.

The function buttons to the right of the list require that a job be selected from the list. To select a job, click on it with the mouse; the selection will be highlighted. You may select only one job at a time.

The fields displayed in the table, as identified in the title line column headings, are shown below.

ID. A unique number assigned to each job.

Media Type. Type of media being used by job. Disk, Tape or Unknown. This field is useful for filtering the Job List.

Type. The type of job. Possible types are:

- Async Mount - An asynchronous mount.
- Default Import - A media import of type "Default".
- Scratch Import – A media import of type “Scratch”.
- Overwrite Import – A media import of type “Overwrite”.
- Export - A media export.
- Deferred Dismount – A cartridge is scheduled for dismount, but the dismount has been delayed in case the cartridge needs to be used by another upcoming job.
- Move - A cartridge being moved to a new PVR.

- Relabel - A cartridge being relabeled.
- Sync Mount - A synchronous mount.
- Tape Check-In - A cartridge being added to the library.
- Tape Check-Out - A cartridge being removed from the library to be placed on the shelf or vault.

Status. The current status of the job. Possible values are:

- Uncommitted – The mount job is a multi-part job that has been started, but the last volumes have not been added, and the mount operation has not been committed.
- Cartridge Wait - The job is waiting for another job to release a cartridge that it needs.
- Drive Wait - The job is waiting for a drive to become available.
- Mount Wait - The job is waiting for a volume to be mounted.
- Mounted - All volumes required for the job are mounted.
- Dismount Pending - Volumes are in the process of being dismounted.
- Aborting - The job is being aborted.
- Inject - Cartridges are being injected into a PVR (for import jobs only).
- Eject - Cartridges are being ejected from a PVR (for export jobs only).
- In Use - All volumes required for the job are mounted and ready for use.
- Deferred Dismount - Dismount for cartridges will be delayed.
- Tape Check-In - The job is waiting for the cartridge to be checked in.
- Tape Check-Out - The job is waiting for the cartridge to be checked out.
- Completed – The job is complete.

Volumes. The number of volumes that are mounted for this job.

Commit Time. The date and time when the job was committed. This will be the same as Request Time for all jobs except asynchronous mounts.

Request Time. The date and time when the job was sent to the PVL

Buttons

Job Info. Click on this button to open the *PVL Request Information* window for the selected job.

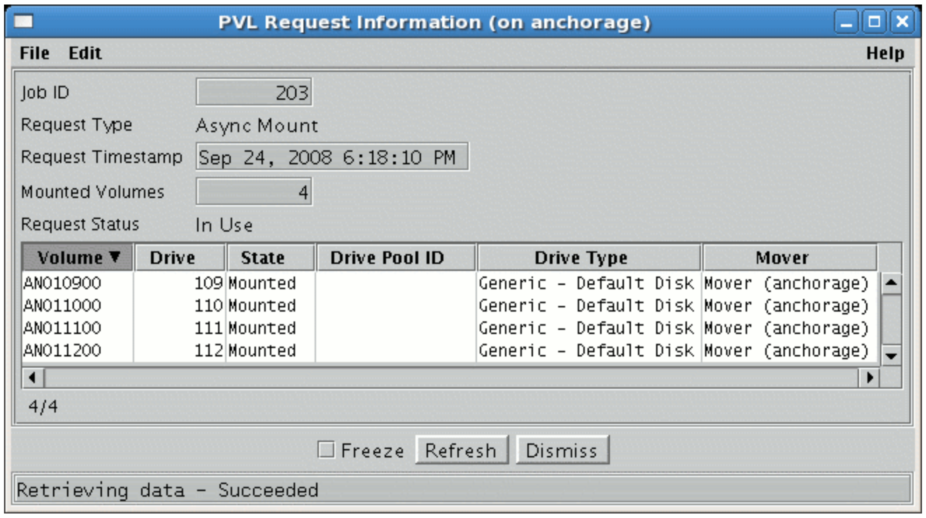
Cancel Job. Click on this button to cancel the selected job. You are prompted to confirm your request before it is actually executed. This will generate a dismount request.

Related Information

Section 8.6.5: *Tape Mount Requests Window* on page 289

Section 8.6.3: *Canceling Queued PVL Requests* on page 288

8.6.2. PVL Request Information Window



Job ID 203
Request Type Async Mount
Request Timestamp Sep 24, 2008 6:18:10 PM
Mounted Volumes 4
Request Status In Use

Volume ▼	Drive	State	Drive Pool ID	Drive Type	Mover
AN010900	109	Mounted		Generic - Default Disk	Mover (anchorage)
AN011000	110	Mounted		Generic - Default Disk	Mover (anchorage)
AN011100	111	Mounted		Generic - Default Disk	Mover (anchorage)
AN011200	112	Mounted		Generic - Default Disk	Mover (anchorage)

4/4

Retrieving data - Succeeded

This window is displayed when the Job Info button is pressed on the *PVL Job Queue* window. It allows you to view the information associated with a PVL job/request.

Field Descriptions

Job ID. The unique number assigned to the job being viewed.

Request Type. The type of job/request. Possible types are:

- Async Mount - An asynchronous mount.
- Default Import - A media import of type default.
- Scratch Import - A media import of type scratch.
- Overwrite Import – A media import of type overwrite.
- Export - A media export.
- Move - A cartridge being moved to a new PVR.
- Relabel - A cartridge being relabeled.
- Sync Mount - A synchronous mount.
- Deferred Dismount - Dismount delayed.
- Tape Check-In - A cartridge being added to the library.
- Tape Check-Out - A cartridge being removed from the library.

Request Timestamp. The date and time when the request was issued to the PVL.

Mounted Volumes. The number of volumes that are currently mounted for this request.

Request Status. The current status of the job/request. Possible values are:

- Aborting - The job is being aborted.
- Cartridge Wait - The job is waiting for another job to release a cartridge that it needs.
- Completed - The job is completed. Once a job is completed, it no longer exists in the PVL job queue, and this window will no longer receive any updates.
- Deferred Dismount - Dismount for cartridges will be delayed.
- Dismount Pending - Volumes are in the process of being dismounted.
- Drive Wait - The job is waiting for a drive to become available.
- Eject - Cartridges are being ejected from a PVR (for export jobs only).
- In Use - All volumes required for the job are mounted and ready for use.
- Inject - Cartridges are being injected into a PVR (for import jobs only).
- Mount Wait - The job is waiting for a volume to be mounted.
- Mounted - All volumes required for the job are mounted.
- Tape Check-In - The job is waiting for the cartridge to be checked in.
- Tape Check-Out - The job is waiting for the cartridge to be checked out.
- Uncommitted - The job has not yet been committed (used for asynchronous mount jobs only).

Volume List. This group of fields shows information on the volumes associated with this request.

Volume. The eight-character volume label.

Drive. The numeric identifier of the drive in which the volume is mounted, or "none" if the volume does not have a drive associated with it.

State. The state of the volume within the context of this request. Possible values are:

- Cart Assigned
- Cart Wait
- Dismount Pending
- Dismounted
- Drive Wait
- Eject
- Inject
- Mount Failed
- Mount Pending
- Mounted
- Reading Label

- Tape Check-In
- Tape Check-Out
- Uncommitted
- Unload Pending

Drive Pool ID. If non-zero the drive pool id will restrict this drive's scheduling to tape requests specifying this value. This field is not applicable for disks.

Drive Type. The HPSS drive type assigned to the drive.

Mover. The name of the mover that owns the device where the volume is mounted. This field will be blank if the volume is not currently mounted.

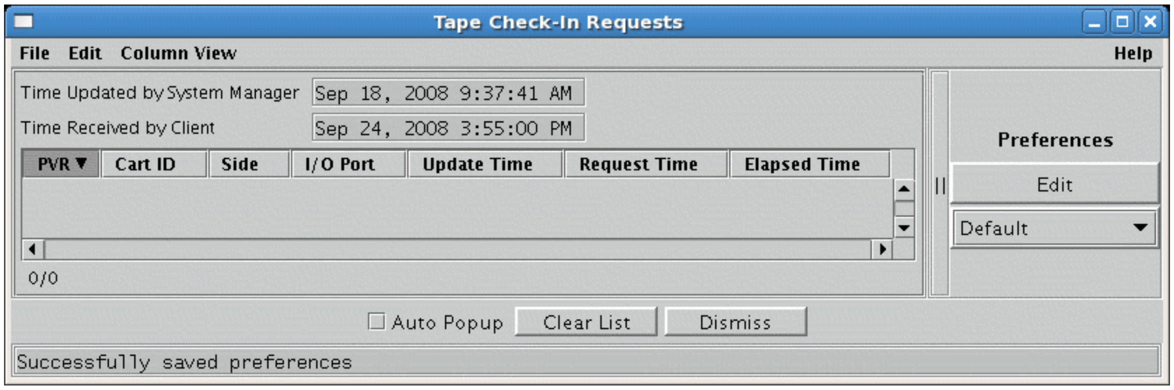
8.6.3. Canceling Queued PVL Requests

PVL requests that cannot be completed due to a hardware or software problem that HPSS cannot handle can be canceled by the system administrator.

Canceling a PVL job will result in the PVL issuing a dismount job request if any of the volumes in the job were mounted. The dismount request causes an unload request to be sent to the controlling Mover. This request may fail if the device is performing an I/O transfer. The PVL will retry the unload command until the I/O completes and the device reservation is freed. Once a volume has been successfully unloaded, all I/O operations directed to the device will fail. Cancelation should therefore be done only after careful consideration of its consequences.

When a deferred dismount job is canceled, the PVL issues dismount requests to all drives loaded with cartridges in deferred dismount state. Canceling a deferred dismount job can be done as often as necessary and does not result in any errors elsewhere in the system.

8.6.4. Tape Check-In Requests Window



Time Updated by System Manager: Sep 18, 2008 9:37:41 AM
Time Received by Client: Sep 24, 2008 3:55:00 PM

PVR ▼	Cart ID	Side	I/O Port	Update Time	Request Time	Elapsed Time

0/0

Successfully saved preferences

When using the shelf-tape feature of HPSS, operators are notified via the *Tape Check-In Requests* window of mount requests for tapes which are stored outside the robot (such as in a vault). Once the requested tapes are inserted back into the robot, tape check-in requests are removed from the list. If a Tape Check-In Request has been displayed, and the corresponding tape has not been checked in, the PVR will begin logging alarms indicating that the appropriate tape has yet to be checked-in. The frequency of these alarms is controlled by the Shelf Tape Check-In Alarm field of the PVR-specific configuration window.

The requests are displayed in chronological order. Each time such a request is received by SSM, it is added to the list, but duplicate requests for the same tape are not displayed. When a message is received indicating that the check-in request has been satisfied, the request is removed from the window. For operator PVRs such check-in requests mean that a tape must be mounted by hand.

The maximum number of tape check-in messages that can be displayed on the *Tape Check-In Requests* window is 100. When this limit is exceeded, some tape check-in notifications will be lost.

If the **Auto Popup** checkbox is selected, this window will automatically reopen whenever a new tape check-in request is received. To open this window manually, select the Monitor menu on the *Health and Status* window, from there select the **Tape Requests** submenu, and from there select the **Check-In** menu item.

The HPSS **shelf_tape** utility may be used to move tapes from off-line storage to a tape library.

Field Descriptions

PVR. The descriptive name of the PVR that owns the cartridge.

Cart ID. The six-character label identifying the cartridge to be inserted into the robot.

Side. The side (partition) number to be mounted. This may be blank.

I/O Port. The descriptive name of the I/O port where the cartridge will be placed.

Update Time. The date and time when the check-in request was last updated.

Request Time. The date and time that the check-in request was first received by the SSM.

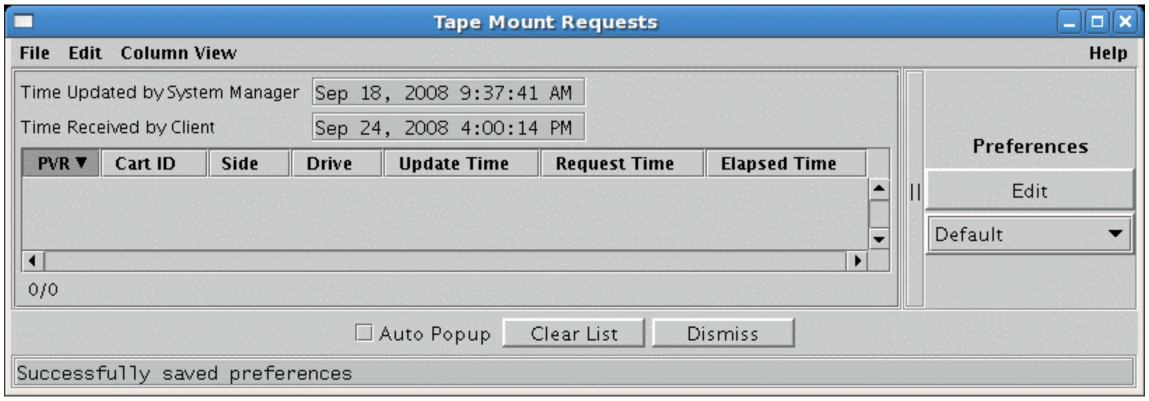
Elapsed Time. The length of time (days-hours:min:sec) that the check-in request has been waiting.

Buttons and Checkboxes

Auto Popup. If ON, this window will be opened automatically if it is not already on the screen when a check-in request is received.

Clear List. Clears the list of check-in requests. Note that this does not cancel any check-in requests, but just removes them from the list. Pending check-in requests will reappear in the window as the PVR periodically retries the check-in of cartridges. This can be useful for removing stale check-in requests that, for some reason, never issued a completion message to SSM. When this button is clicked from any SSM session, the *Tape Check-In Requests* windows on all SSM sessions will be cleared.

8.6.5. Tape Mount Requests Window



Time Updated by System Manager: Sep 18, 2008 9:37:41 AM

Time Received by Client: Sep 24, 2008 4:00:14 PM

PVR	Cart ID	Side	Drive	Update Time	Request Time	Elapsed Time

0/0

Successfully saved preferences

The *Tape Mount Requests* window displays tapes which need to be mounted in a drive. All HPSS tape mount requests, including both robotic and operator tape mounts, will be displayed in the window. For operator PVRs, such mount requests mean that a tape must be mounted by hand. When mount requests for robotic PVRs do not disappear from the window in a timely manner, it can be an indication of hardware or other problem in the robot.

Outstanding tape mounts, along with the associated PVR, are displayed in chronological order. Each time a tape mount request is received by SSM, it is added to the list, but duplicate requests for the same cartridge are not displayed. When a message is received indicating that the mount request has been satisfied, the request is removed from the window.

The window may display an old mount request at times, due to lost notifications. Click on the **Clear List** button to refresh the window.

When a new mount notification is received, SSM may bring up the *Tape Mount Requests* window automatically if it is not currently displayed. This behavior is disabled by deselecting the **Auto Popup** checkbox. To open the window manually, select the Monitor menu on the *Health and Status* window, from there select the **Tape Requests** submenu, and from there select the **Mount** menu item..

Field Descriptions

PVR. The descriptive name of the PVR that owns the cartridge.

Cart ID. The six-character label identifying the cartridge to be mounted.

Side. The side (partition) number to be mounted. This may be blank.

Drive. The drive address if a specific drive was requested for the mount. In most cases, a drive is not specified, and this is blank.

Update Time. The date and time when the mount request was last updated.

Request Time. The date and time that the mount request was first received by the SSM.

Elapsed Time. The length of time (days-hours:min:sec) that the mount request has been waiting.

Buttons and Checkboxes

Auto Popup. If ON, this window will be opened automatically if it is not already on the screen when a mount request is received.

Clear List. Clears the list of mount requests. Note that this does not cancel any mount requests, but just removes them from the list. Pending mounts will reappear in the window as the PVR periodically retries the mounts. This can be useful for removing stale mount requests that, for some reason, never issued a completion message to SSM. When this button is clicked from any SSM session, the *Tape Mount Requests* windows on all SSM sessions will be cleared.

8.6.6. Administrative Tape Dismounts

HPSS provides the ability to administratively command a tape dismount in unusual situations, such as when an administrator wants a tape dismounted without waiting for HPSS to do so, or when HPSS is unable to do so due to a failure.

Administrative dismounts should only occur in exceptional error circumstances. Dismounting a tape may cause the operations being performed on that tape to fail. A library dismount operation is unlikely to occur while a mover has the device reserved for I/O.

Use the *Devices and Drives* list window to issue a dismount for the desired device/drive entry. A confirmation window will pop up requiring you to confirm the dismount request.

8.7. New Storage Technology Insertion

As more advanced storage technology becomes available, there may be a need to replace used storage media with newer volumes. HPSS provides a procedure for making this change. The procedure retires the old volumes, introduces new volumes, moves the storage segments from the old to the new, and then removes the old volumes.

The volume retirement feature is provided to facilitate new technology insertion. HPSS will not select a retired volume to receive a new storage segment. In every other way, retired volumes behave normally. Storage segments on retired disk volumes can be read and written normally. Storage segments on retired tape volumes can be read normally. Attrition of segments through the usual processes of file deletion and migration will cause retired volumes to gradually become empty.

A retired volume has the characteristic that its VV Condition is something other than RWC, and cannot be changed to RWC. New storage segments can be created on a volume only when the VV Condition is RWC, so new segments are not created in retired volumes.

To replace old technology volumes with new, the following outline serves as a guide, but the administrator should be aware that the procedure will probably need to be tailored to the specific circumstances of the change. This procedure applies to both tape and disk volumes:

1. Change the storage class definition by deleting and then recreating it using the same storage class ID. The new storage characteristics must reflect the characteristics of the new storage technology volumes. In addition, they should be compatible with other storage levels in the hierarchy.

   Changes made to the storage class definition do not affect volumes that exist when the changes are made. The characteristics of established volumes stay the same throughout their lifetimes. Changes made to the storage class only affect the characteristics of volumes created after the change is made.

2. Stop and restart the Core Servers and Migration Purge Servers that use the modified storage class. These servers cache storage class definitions in memory and should have the opportunity to

refresh their caches.

3. Use the **retire** utility program to retire the old technology volumes. r**etire** can accept a list of PVs to retire, or can retire all volumes in a storage class that are in a specified Condition. **dump_sspvs** can be used to create a list of PVs for **retire** to process. Creating a list of PVs for this step and the following steps is the recommended procedure. If the number of volumes to retire is small, they can be retired manually, using SSM, by changing the Condition of the volume and setting the Retire flag using the *Core Server Tape Volume* (or *Core Server Disk Volume*) window. Retiring a volume is sufficient to prevent it from receiving new storage segments. It is not necessary to change the Condition of tape volumes to EOM; RO Condition will prevent the tapes from being written and is reversible. Tapes changed to EOM condition cannot be changed to any other Condition other than Down.
4. Import and create new storage technology volumes. These volumes will take on the characteristics of the re-defined storage class. The administrator should probably create only one volume at first in order to check its characteristics. If they are not as desired, the volume can be deleted, and steps one and two can be repeated. It is not necessary to create all the new technology volumes at once. Only as many as are required to meet the immediate needs of the system need to be created. Keep in mind that when storage resources are created, they are immediately available for use by the system.
5. Either allow the natural attrition process to take place if time is not an issue, or speed up the replacement process by using the **repack** utility to repack all retired volumes in the storage class. This step is frequently a long process and may require regular attention from the administrator. Use **dump_sspvs** to monitor the progress of the repack process and to determine that all retired volumes have been emptied. Note that **repack** can accept a list of volumes to process, and the list can be submitted to **repack** repeatedly without harm.
6. Invoke the **remove** utility to remove all retired and empty volumes. This utility will delete all resources associated with these volumes and export them from HPSS. Note that **remove** accepts a list of PVs to remove.

Chapter 9. Logging and Status

9.1. Logging Overview

The purpose of logging is to record events of interest that occur in HPSS in the sequence they occur to support diagnostic research.

HPSS provides eight log message types:

- Alarm
- Event
- Status
- Debug
- Request
- Security
- Accounting
- Trace

The purpose of each of these log message types is described later in this chapter.

Log messages are deposited in four places. Certain high priority messages are transmitted to SSM and displayed on the Alarms and Events window. This allows the administrator to see events unfolding in close to real time. All log messages are coded in a binary format and written to one of two central log files. The Log Daemon writes these files alternatively. Also, the Log Client on each host writes a large circular ASCII file with log messages just for that host as they are received. This file makes it easy for the administrator to copy the file and quickly search it for an event of interest using ordinary text processing tools. Finally, if HPSSLOGGER is set, HPSS will log to the location specified by HPSS LOGGER (syslog or stdout) if the logger isn't running/available.

The HPSS logging facility is comprised of two types of servers, Log Clients and the Log Daemon. Log Clients run on each HPSS node. A Log Client receives log messages from each HPSS server running on its node and filters those messages based on the configured log policies. Messages that pass the filter are logged locally based on the Log Client’s configuration and are then sent to the Log Daemon. The Log Daemon checks the log policies to see which messages are to be displayed in the *Alarms and Events* SSM window before entering them into the central log.

For useful logging to take place, it is important that the log policies, Log Clients and Log Daemon are all configured and managed properly.

The HPSS logging system includes these components:

1. Log Policies (Section 9.2: *Log Policies* on page 295)
2. Local logs (Section 9.5 *Error: Reference source not found* on page 301)
3. The Central Log (Section 9.3: *Managing the Central Log* on page 299)
4. Log Clients and Log Daemon (Section 5.1.4: *Log Client Specific Configuration* on page 100 and Section 5.1.1: *Log Daemon Specific Configuration* on page 101)

5. The SSM *Alarms and Events* window (Section 5.2.2.3: on page 151)

A standard configuration for logging services is usually set by the administrator during the HPSS system configuration. Specialized configurations can be set up and used to temporarily (or permanently) provide more or less logging for site-specific or shift-specific operational situations. Increasing the level of logging may slow down overall system operation due to the overhead of extra messages, and decreasing the amount of logging may eliminate certain log messages that could prove useful in a postmortem analysis of problem situations. The logging configuration can be tailored as needed to satisfy site or shift requirements.

Each SSM graphical user session (hpssgui) and command line session (hpssadm) can write a session log file of the SSM error and informational messages issued during the session. These are not the same as the alarm, event, and other HPSS log messages described above and are not described in this chapter. See the '-S' option in the hpssgui and hpssadm man pages for details.

9.2. Log Policies

For any HPSS server or utility which uses the HPSS logging facility, a log policy can be configured to specify which log message types will be sent to the HPSS logs and which message types will be sent to SSM for display. Log policies are configured by Server Name, which is displayed on each HPSS Server's *Server Configuration* window.

If no log policy is configured for a server, the server will use the Default Log Policy. If no default is defined, only Alarm and Event message types will be logged.

After changing any log policy information, including the Default Log Policy, the appropriate Log Clients must be reinitialized to make the changes effective. The Log Clients associated with a particular policy are the ones which execute on the same hosts as the servers to which the policy applies. For example, if you modified only policies for servers which execute on host A, then only the Log Client which runs on host A needs to be reinitialized. The Mover log policy is handled differently. If the Mover log policy is modified, the Mover itself must be reinitialized in order for the policy changes to take effect.

9.2.1. Creating a Log Policy

A log policy can be created for any unique name which is typed into the Descriptive Name field on the Logging Policy window, regardless of whether or not the name refers to any actual server or utility.

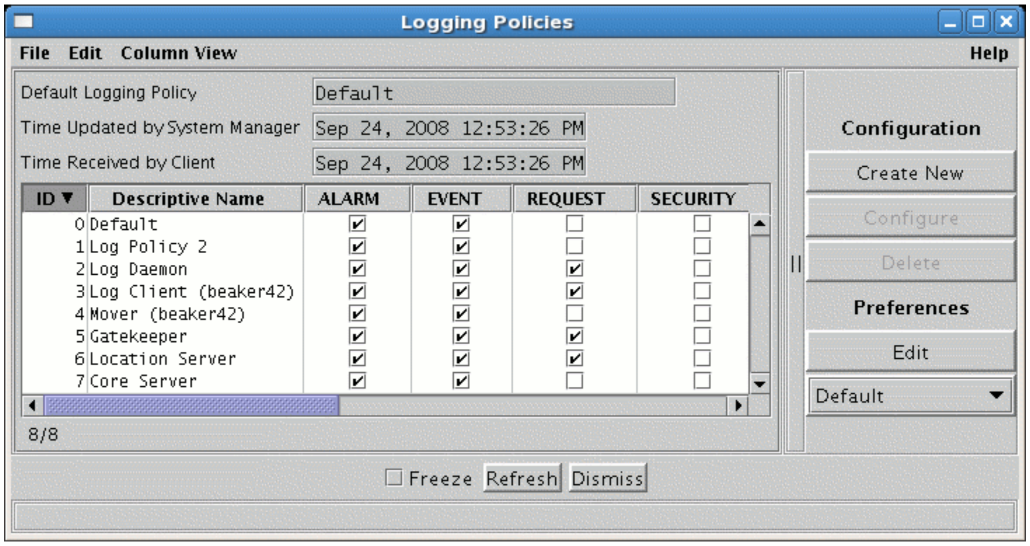
A new log policy can be created and managed using the *Log Policy* window or via the **Log Policy** tab of of an HPSS Server's *Server Configuration* window (Section 5.1.1.1: *Log Policy* on page 95). This *Log Policy* window can be accessed from the *Logging Policies window* (Section 9.2.2: *Logging Policies Window* on page 296). From this window, a log policy can be created for any unique name which is typed into the Descriptive Name field on the Logging Policy window, regardless of whether or not the name refers to any actual server or utility.

A server's logging policy is determined according to the following precedence rules:

- If a server-specific logging policy is configured for a server, that policy is used for the server.
- If no server-specific logging policy is configured for a server, but a default logging policy is configured, the default policy will be used for the server.

- If no server-specific logging policy is configured for a server and no default logging policy is configured, only Alarm and Event messages will be logged.

9.2.2. Logging Policies Window



Default Logging Policy: Default
Time Updated by System Manager: Sep 24, 2008 12:53:26 PM
Time Received by Client: Sep 24, 2008 12:53:26 PM

ID ▼	Descriptive Name	ALARM	EVENT	REQUEST	SECURITY
0	Default	✓	✓	☐	☐
1	Log Policy 2	✓	✓	☐	☐
2	Log Daemon	✓	✓	✓	☐
3	Log Client (beaker42)	✓	✓	✓	☐
4	Mover (beaker42)	✓	✓	☐	☐
5	Gatekeeper	✓	✓	✓	☐
6	Location Server	✓	✓	✓	☐
7	Core Server	✓	✓	☐	☐

8/8

This window is used to manage all the log policies in the HPSS system.

To create a new log policy, click on the Create New button. To configure an existing log policy, select the policy from the list and click on the Configure button. When creating or configuring a log policy, the *Logging Policy* window will appear. To delete an existing log policy, select the policy from the list and click on the Delete button.

Log policies for servers can also be created using a server's configuration window which can be accessed via the *Servers* window (Section 5.1.1.1: *Log Policy* on page 95). Log policies for a server can also be deleted from the same window by pressing the **Use Default Log Policy** button.

Field Descriptions

Default Logging Policy. The descriptive name of the default logging policy. This policy will apply to all servers which do not have their own policy defined. This field is non-editable but can be configured by modifying the *Global Configuration* window (Section 4.1: *Global Configuration Window* on page 72).

Log Policy List table columns: For details on the fields listed for each log policy, refer to the *Logging Policy* window (Section 9.2.2.1 *Logging Policy Configuration Window* on page 297.

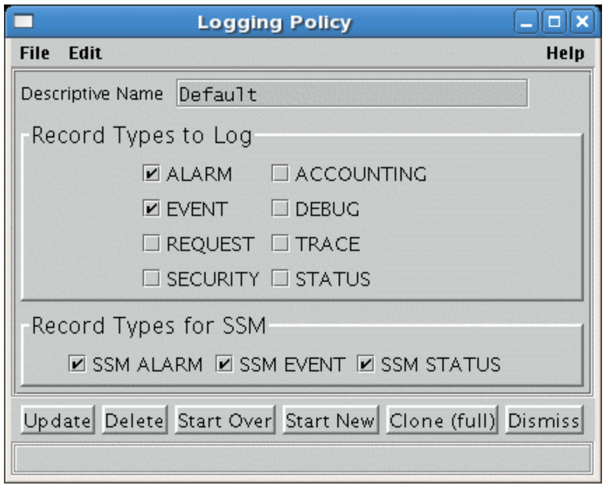
Configuration Buttons

Create New. Opens a *Logging Policy* window containing default values for a new policy.

Configure. Opens the selected log policy configuration for editing.

Delete. Deletes the selected log policy(s).

9.2.2.1. Logging Policy Configuration Window



Descriptive Name Default

Record Types to Log

☑ ALARM ☐ ACCOUNTING
☑ EVENT ☐ DEBUG
☐ REQUEST ☐ TRACE
☐ SECURITY ☐ STATUS

Record Types for SSM

☑ SSM ALARM ☑ SSM EVENT ☑ SSM STATUS

The *Logging Policy* window is used to manage a log policy.

When creating a new log policy, the Descriptive Name field will be blank, and a set of default options will be selected. If there is a Default Logging Policy defined, the default log options will match those in the global configuration's Default Logging Policy. If there is no Default Logging Policy defined in the global configuration, then the Alarm, Event, SSM Alarm, SSM Event and SSM Status log options will be enabled and the rest will be disabled.

For logging of a record type to be enabled, the checkbox located to the left of the log record type must be selected. The log state for each record type can be toggled by clicking on the checkbox.

After creating or modifying a Logging Policy, including the Default Logging Policy, reinitialize the appropriate Log Clients to make the changes effective. For example, if you only modified policies for servers which execute on host A, then only the Log Client which runs on host A needs to be reinitialized. Mover log policies are managed differently; if you modify the log policy for a Mover, that Mover itself must be reinitialized for the new log policy to take effect.

Field Descriptions

Descriptive Name. The descriptive name of the HPSS server or utility to which this log policy applies. The name must match exactly to the descriptive name used by the server or utility; otherwise the Default Logging Policy will be used. The Descriptive Name is case sensitive. For a new policy, this field will be blank and the user must enter a server name. Since it is easy to enter a mis-matched Descriptive Name, it is recommended that the log policy be created via the server's configuration window using the Log Policy tab where the Descriptive Name will be automatically filled in.

Record Types to Log. Record types that are to be logged for the specified server. If a checkbox is selected for a record type, that type will be recorded by the Log Client. The record types are:

- ALARM. A message reporting a high-level error condition of interest to the administrator. It is recommended that this always be selected.

- EVENT. An informational message (e.g., subsystem initializing, subsystem terminating) about a significant occurrence in the system that is usually not an error. It is recommended that this always be selected.
- REQUEST. A message which reports the beginning and ending of processing of a client request in a server. It is recommended that this type not be selected except for short periods as an aid to isolating a problem.
- SECURITY. A message reporting security related events (e.g., authorization failures). It is recommended that this always be selected.
- DEBUG. A lower-level error message. For troubleshooting, these messages are important record types because they provide more detailed information about the root cause of an error. It is recommended that this always be selected.
- TRACE. A message for tracing any type of entry/exit processing flows. It is recommended that this type not be selected except for short periods as an aid to isolating a problem.
- STATUS. A message reporting the interim progress of a long-running task.
- ACCOUNTING. A message which logs accounting information.

It is recommended that at least the Alarm, Event, Security and Status record types be selected for all servers while they are running normally. Core Servers and Movers should also log Debug messages, and additional record types for debugging purposes should be selected for servers experiencing problems. If system performance is being degraded by excessive logging, first turn off Trace, Debug and Request record types for all servers. Other record types can be turned off as necessary. However, the HPSS administrator may lose useful debugging information when problems occur.

To facilitate diagnosis of data transfer problems, it is recommended that Debug logging always be turned on for all Movers.

Record Types for SSM. Record types that are to be sent to SSM for display. If a box is selected for a record type, that type will be sent to the SSM for display in the *Alarms and Events* window (Section 5.2.2.3: on page 151). The allowed types are a subset of the allowed Record Types to Log described above:

- SSM ALARM
- SSM EVENT
- SSM STATUS

It is recommended that Alarm and Event record types be sent to SSM.

9.2.3. Changing a Log Policy

The log policy for a configured server may be accessed from the Log Policy tab of the *Server Configuration* window for that server (Section 5.1.1.1: *Log Policy* on page 95), or from the *Logging Policies* window (Section 9.2.2: *Logging Policies Window* on page 296).

A server's log policy can be modified to control the volume of messages to the chosen logging destinations. Typically, during normal operations, the level of logging may be decreased to only Alarm, Event, and Security to reduce overhead. However, while tracking an HPSS problem, it may be desirable to include more log message types such as Debug, Request and Trace to obtain more information. As stated previously, it is recommended that Trace not be selected except for short periods as an aid to isolating a problem.

After a server's log policy has been modified, the Log Client running on the same node as the server must be reinitialized for it to reread the new log policy.   If the server is a Mover, then that Mover itself must be reinitialized in order for the new log policy to take effect.

9.2.4. Deleting a Log Policy

To delete a log policy, select the log policy from the *Logging Policies* window (Section 9.2.2: *Logging Policies Window* on page 296) and click the Delete button. A log policy can also be deleted by opening the *Logging Policy* window (Section 9.2.2.1: *Logging Policy Configuration Window* on page 297) for the appropriate log policy and clicking the Delete button.

A third way to delete a log policy is to open the Log Policy tab of the *Server Configuration* window for that server and push the Use Default Log Policy button. This removes the unique log policy configured for that server and makes it revert to using the default policy.

Before deleting a log policy, ensure that no servers are configured to use it. Otherwise, the log policy deletion could result in loss of important log messages. After deleting a log policy, the Log Clients supporting any servers that were using the policy must be reinitialized or restarted. If the log policy was used by a Mover, then that Mover itself must be reinitialized or restarted in order for the Mover to stop using the deleted policy.

9.3. Managing the Central Log

This section describes configuring and viewing the central log file. Note that the central log is located on the node with the Log Daemon and is in binary format.

9.3.1. Configuring Central Log Options

The central HPSS log is managed by the Log Daemon and consists of two log files, logfile01 and logfile02. These log files can be controlled by changing the Archive Logfiles and the Switch Logfiles flags on the *Log Daemon Specific Configuration* window (Section 5.1.1: *Log Daemon Specific Configuration* on page 101). The Archive Logfiles flag dictates whether the log files will be automatically archived when they are filled. The Switch Logfiles flag dictates whether a switch to the second log file should be done if the archive of the second log file has not yet completed. The administrator needs to consider whether archiving of the log files is needed and then set these flags appropriately.

In addition to modifying the logging configuration to control logging, the HPSSLOG_SHMKEY, HPSSLOG, and HPSSLOGGER environment variables described in the HPSS Installation Guide, Section 5.6: Define HPSS Environment Variables can also be used.

9.3.2. Viewing the Central Log (Delogging)

HPSS provides the ability to retrieve and examine HPSS log records as a means of analyzing the activity and behavior of HPSS. The retrieval and log record conversion process is referred to as “delogging.” Delogging is the process of retrieving specific records from the HPSS central log files (which is in binary format), converting the records to a readable text format, and sending the resulting text to a local UNIX file.

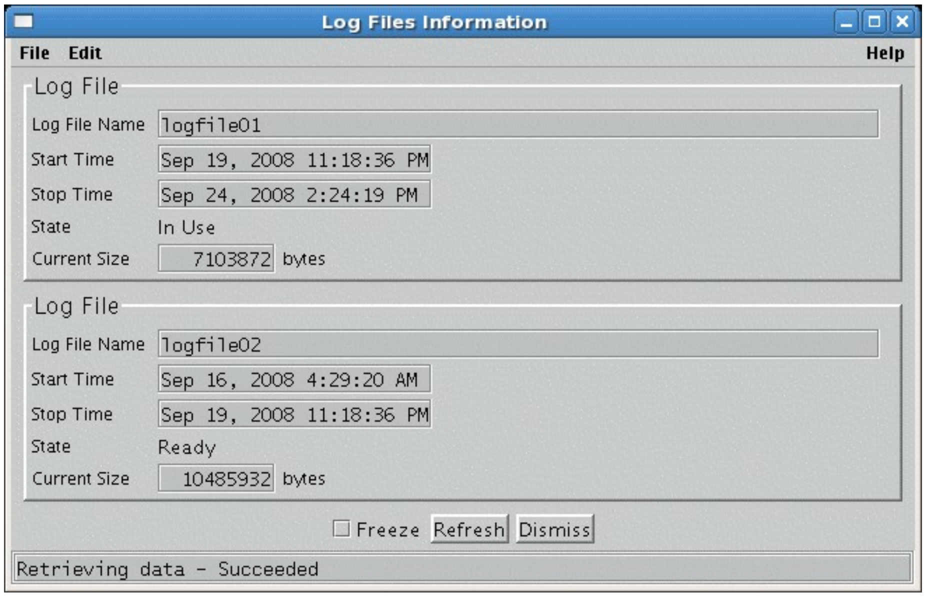
To effectively view log records, the user should know what kinds of activities or behaviors are of interest. Since it is possible to configure log policies to include almost all activity in HPSS, in most cases it will be easier to examine and analyze logs by filtering out any records that are not of interest. The **hpss_delog** utility allows selective filtering of log records based on start time, end time, server descriptive name, log record type, user name, principal, authentication mechanism, and authenticator name.

The central log file names available for delogging are logfile01 and logfile02. These files reside in the directory name specified in the Log Daemon configuration entry. To determine whether a message of interest is in a log file, open the associated *Log File Information* window (Section 9.4: *Log Files Information* on page 300) to obtain the start and end times of the log file and compare them with the message timestamp.

If log files that have been archived to HPSS are to be delogged, they will reside in a date-based directory structure in HPSS: /log/yyyy/mm/dd. To determine the log times for an archived log file, list the log files. The timestamp of the time when the file was archived to HPSS is appended to the end of the log file name. The format of the archive file names is logfile01_yyyymmddhhmm or logfile02_yyyymmddhhmm.

Use the **hpss_delog** utility to convert the logfile01, logfile02, or archived log file retrieved from HPSS. Please see the **hpss_delog** man page for full usage information and examples.

9.4. Log Files Information



Log File
Log File Name: logfile01
Start Time: Sep 19, 2008 11:18:36 PM
Stop Time: Sep 24, 2008 2:24:19 PM
State: In Use
Current Size: 7103872 bytes

Log File
Log File Name: logfile02
Start Time: Sep 16, 2008 4:29:20 AM
Stop Time: Sep 19, 2008 11:18:36 PM
State: Ready
Current Size: 10485932 bytes

Retrieving data - Succeeded

The information is acquired from the HPSS Log Daemon.

The *Log Files Information* window provides information about the HPSS central log files. Log file information includes the state of each log file, the current size of each log file in bytes, the time at which each log file was marked in use, the time at which each log file was last active, and the log file names.

Field Descriptions

For each of the two central log files, the following fields are displayed:

Log File Name. The name of the log file. This will be logfile01 or logfile02.

Start Time. The time and date when the logfile was marked In Use.

Stop Time. The time and date when the logfile was last In Use.

State. The state of the logfile. Possible states are:

- In Use - the file is currently active; it is the file to which new log messages are being written
- Ready - the file is ready for use, but not currently active
- Archive - the file is marked for archiving
- Archiving - the file is being archived
- Invalid - the file is not a valid logfile

Current Size. The physical size of the logfile in bytes.

9.5. Managing Local Logging

This section describes configuring and viewing the local log file. A local log (local.log) is located on every node where a Log Client resides.

9.5.1. Configuring Local Logging Options

The “Log Messages To:” field in the *Logging Client* specific configuration window (Section 5.1.4: *Log Client Specific Configuration* on page 100) can be modified to control the destinations for the messages logged by the HPSS servers running in a node. This parameter consists of a set of options that apply to the local logging. These options are:

- **Log Daemo**n - Send log messages to the central log.
- **Local Logfile** - Send log messages to the local log file. This file is written in round-robin fashion, and new log messages will eventually overwrite older ones.
- **Syslog** - Send log messages to the syslog.
- **Stdout** - Send log messages to standard output.

It is not recommended to specify both the **Local Logfile** and the Standard Output (**Stdout)** options, but any other combination of these options is supported. If neither the **Local Logfile** option nor the **Syslog** option is specified, no local logging will occur to a local log file. If the Log Daemon option is not specified, no messages from the HPSS servers running on the same node as the Log Client will be written to the central log (logfile01/logfile02).

The administrator should choose the local logging options that will satisfy the site’s need. After a Log Client’s configuration parameter has been changed, the Log Client must be reinitialized.

*The **syslog** option should be used with care. The syslog file will grow without bound until deleted/ truncated, or until the file system runs out of space.*

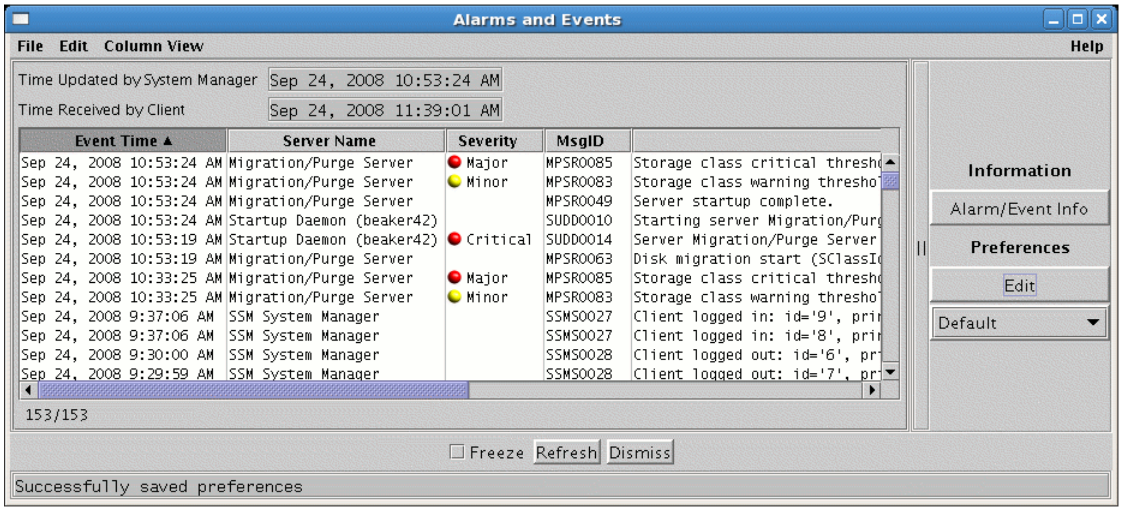
9.5.2. Viewing the Local Log

The local log file is an ASCII text file and can be viewed as such. The name of the local log file is specified in the Log Client’s configuration entry.

9.6. Managing SSM Alarms and Events

This section describes how to manage SSM alarms and events.

9.6.1. Alarms and Events Window



Time Updated by System Manager: Sep 24, 2008 10:53:24 AM

Time Received by Client: Sep 24, 2008 11:39:01 AM

Event Time ▲	Server Name	Severity	MsgID	
Sep 24, 2008 10:53:24 AM	Migration/Purge Server	🔴 Major	MPSR0085	Storage class critical thresh
Sep 24, 2008 10:53:24 AM	Migration/Purge Server	🟡 Minor	MPSR0083	Storage class warning thresho
Sep 24, 2008 10:53:24 AM	Migration/Purge Server		MPSR0049	Server startup complete.
Sep 24, 2008 10:53:24 AM	Startup Daemon (beaker42)		SUDD0010	Starting server Migration/Pur
Sep 24, 2008 10:53:19 AM	Startup Daemon (beaker42)	🔴 Critical	SUDD0014	Server Migration/Purge Server
Sep 24, 2008 10:53:19 AM	Migration/Purge Server		MPSR0063	Disk migration start (SClassI
Sep 24, 2008 10:33:25 AM	Migration/Purge Server	🔴 Major	MPSR0085	Storage class critical thresh
Sep 24, 2008 10:33:25 AM	Migration/Purge Server	🟡 Minor	MPSR0083	Storage class warning thresho
Sep 24, 2008 9:37:06 AM	SSM System Manager		SSMS0027	Client logged in: id='9', pri
Sep 24, 2008 9:37:06 AM	SSM System Manager		SSMS0027	Client logged in: id='8', pri
Sep 24, 2008 9:30:00 AM	SSM System Manager		SSMS0028	Client logged out: id='6', pr
Sep 24, 2008 9:29:59 AM	SSM System Manager		SSMS0028	Client logged out: id='7', pr

153/153

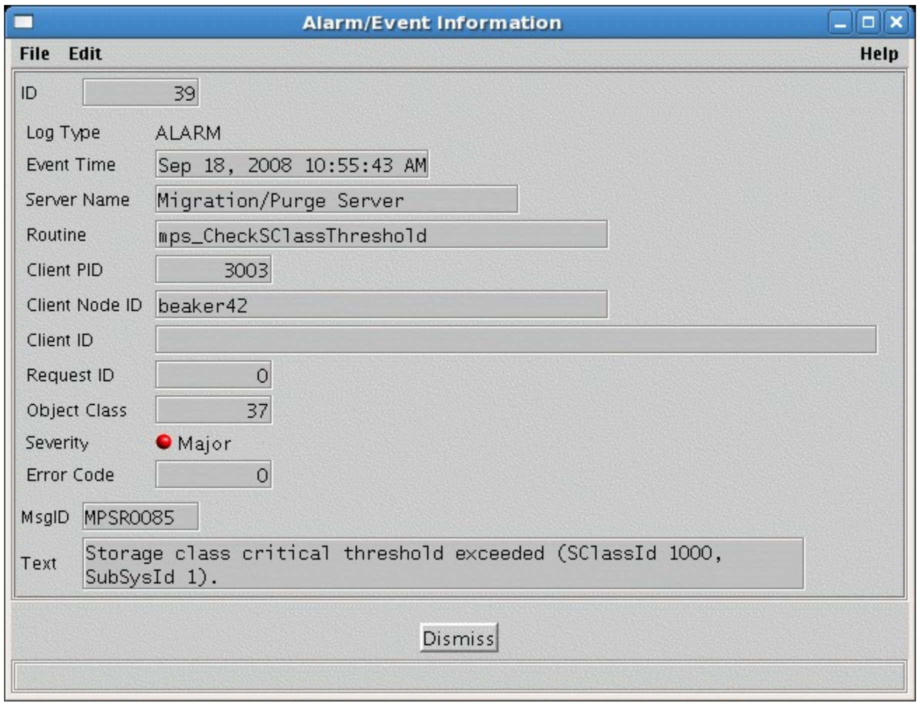
Successfully saved preferences

This window displays a number of the most recent alarm and event messages which have been received by SSM. It also allows you to view individual messages in greater detail by selecting the message and pressing the Alarm/Event Info button to bring up the *Alarm/Event Information* window.

Field Descriptions

This list displays a column for each field shown on the *Alarm/Event Information* window. See Section 9.6.2: *Alarm/Event Information* for the field descriptions.

9.6.2. Alarm/Event Information



ID 39
Log Type ALARM
Event Time Sep 18, 2008 10:55:43 AM
Server Name Migration/Purge Server
Routine mps_CheckSClassThreshold
Client PID 3003
Client Node ID beaker42
Client ID
Request ID 0
Object Class 37
Severity ● Major
Error Code 0
MsgID MPSR0085
Text Storage class critical threshold exceeded (SClassId 1000, SubSysId 1).

This window displays all the details of the alarm or event selected from the *Alarms and Events* window.

Field Descriptions

ID. A sequence number assigned to the log message by SSM. This ID is not used outside of SSM.

Log Type. General class of the message. May be either “Alarm” or “Event”.

Event Time. The date and time the message was generated.

Server Name. Descriptive name of the HPSS server or utility that logged the message.

Routine. Name of the function that was executing when the message was logged.

Client PID. Process ID of the process that logged the message.

Client Node ID. Name of the host where the message was generated.

Client ID. The userid of the process which generated the message.

Request ID. Request ID generated by the process logging the message.

Object Class. Process-specific ID number assigned by the process logging the message.

Severity. For events, this field will be blank. For alarms, it will be one of the following:

- Warning
- Minor
- Major

- Critical
- Indeterminate
- Cleared

These may be accompanied by a color status indicator:

- 🔴 (Red) - Critical or Major alarm
- 🟡 (Yellow) - Minor or Warning alarm
- None – Events and other alarm types

Error Code. The error code associated with the problem underlying the message.

MsgID. The 8-character ID code for this message, consisting of a 4-character mnenomic identifynig the type of server or subsystem which issued the message followed by a 4-digit message number. (The message number is not the same as the Error Code field.) Message IDs may be used to look up messages in the *HPSS Error Message Manual*.

Text. The text of the alarm or event message. Only 240 characters are displayed. Look at log to get entire message.

9.6.3. Diagnosing HPSS Problems with Alarms and Events

Events displayed on the *Alarms and Events* window provide an indication that a significant event has occurred in a server and the event may be of interest to the administrator. An alarm, however, indicates that a server has detected an abnormal condition. The user should investigate the problem as soon as possible to ensure timely resolution. The user may use the information provided by the alarm to obtain further information on the problem as follows:

- Use the alarm message number to look up the alarm information in the *HPSS Error Messages Reference Manual*. For each documented message, the manual provides more detailed information on the problem and its possible source. The manual also provides recommendations on how to resolve the problem.
- If additional information is needed, use the alarm timestamp to delog the HPSS central log for the log messages received prior to and after the problem is reported. In addition to obtaining the messages logged by the server in question, it may be necessary to obtain log messages from other HPSS servers that interface with the server. Refer to Section 9.3.2: *Viewing the Central Log (Delogging)* on page 300 for more information on delogging the HPSS log messages.

9.6.4. Controlling SSM Log Message Handling

This section describes the options available to control SSM handling of HPSS log messages.

This does not include the SSM session log. See the -S option on the hpssgui or hpssadm man page for a discussion of SSM session logging.

9.6.4.1. Controlling the System Manager Log Message Cache

By default, the SSM System Manager stores the alarms, events, and status messages it receives in an internal memory cache. This cache can be configured instead to be kept in a disk file by defining the environment variable HPSS_SSM_ALARMS with the desired name of the cache file. The default for HPSS_SSM_ALARMS is defined in hpss_env_defs.h as NULL. SSM will revert to the internal memory cache if it cannot access the specified cache file for any reason.

The site may set the HPSS_SSM_ALARMS environment variable to any UNIX file that has read/write access for user root on the machine where the SM is to be run (since the SM runs as user root). A good path might be /var/hpss/log/alarm_buffer to keep the log files all in one place.

Using a disk cache makes SSM alarms and events "persistent"; that is, if the System Manager is restarted, SSM will be able to display all old messages still in the disk cache. Caching the alarms and events in memory means the alarm cache starts out empty each time the System Manager is restarted.

By default, 2000 messages are cached before the oldest are discarded. This value can be modified by setting the HPSS_SSM_ALARMS_DISPLAY environment variable to the desired number of messages to be retained and restarting the System Manager.

Once an old message is discarded, it cannot be accessed anymore from the *Alarms and Events* window. Older messages can be accessed by the delog utility.

Since the messages are cached and managed by the System Manager, all SSM users connected to the System Manager will see the same messages. However, preferences selected on the *Alarm and Event Preferences* window are applied on a user-by-user basis.

Messages can be delivered to the System Manager and stored in the cache only while the SSM System Manager is running. Any messages generated while the System Manager is down will not be accessible from the HPSS *Alarms and Events* window (but they should be accessible from the delog utility). This means that there can be gaps in the list displayed in the *Alarms and Events* window if the System Manager is shut down and restarted while the rest of HPSS is up.

9.6.4.2. Controlling Log Messages Displayed by hpssgui and hpssadm

The hpssgui and hpssadm programs each keep an internal cached copy of the Alarm and Event list. The list is maintained by regular polling requests to the System Manager. Please review and reference the hpssgui and hpssadm man pages for further details on the settings discussed in this section.

The polling interval is set by the HPSSSSM_UI_ALARM_RATE environment variable, which may be overridden by the "-A" option to the hpssgui or hpssadm startup script. The default is to poll the System Manager every 5 seconds for new log messages.

Each polling request returns only "new" messages, those which have not already been retrieved by the hpssgui or hpssadm in a previous polling request. If there are no new messages in the System Manager log cache, no messages are returned.

By default, the maximum number of messages retrieved by each polling request is 2000. If there are more than 2000 new messages in the System Manager log message cache, only the 2000 newest will be returned. This means that if new messages are arriving rapidly at the System Manager and the hpssgui or hpssadm polling interval and maximum number of messages values are not set appropriately, there can be gaps in the hpssgui or hpssadm Alarm and Event list.

The value for the maximum number of messages to return by each polling request may be redefined by the HPSS_SSM_ALARM_GET environment variable or by the "-G" option to the hpssgui or hpssadm startup script. The command line option takes precedence over the environment variable. This option could be used, for example, to reduce the size of each data request on a slow network.

The internal cached Alarm and Event list is displayed by the hpssadm program by means of its "alarm list" command. This command has a "-c" option to specify how many of the most recent log messages in the internal copy to display. If more messages are requested than exist in the internal list, the full internal list is displayed. See the hpssadm man page for details.

The internal cached Alarm and Event list is displayed by the hpssgui in the Alarm and Events window as described in Section 5.2.2.3: on page 151. The maximum number of log messages displayed in the window is set by the HPSS_SSM_ALARM_DISPLAY environment variable, which may be overridden by the "-N" option to the hpssgui startup script. The default is to display at most 2000 messages in the list.

If the maximum number of messages to retrieve (HPSS_SSM_ALARM_GET or -G option) is greater than the maximum number of messages to display (HPSS_SSM_ALARM_DISPLAY or -N option), it will be reset to the same value as the maximum number of messages to display. This is true for both the hpssgui and the hpssadm, although the hpssadm does not use the maximum number of messages to display value for anything else.

If new messages are arriving regularly, the messages displayed by the hpssgui Alarms and Events window will constantly be moving downward at set intervals. This can make it difficult to select a specific message if the window is updating frequently. This problem can be minimized by being careful to click on a message immediately after an update cycle, or it can be eliminated entirely by freezing the display by checking the **Freeze** checkbox at the bottom of the Alarms and Events window.

If many messages are arriving very rapidly, or if you leave the **Freeze** button on, it is possible for the display to become stale. A stale message is one which is still visible, but which has been discarded from the cache. If you click on such a message, an error box will appear, telling you that the selected message is no longer available.

Messages are displayed in the order in which they are received by the System Manager. They are not sorted by the event time displayed with each message, so if many messages are generated in a short period of time, it is possible to see messages which are displayed slightly out of chronological order.

Each hpssgui user can filter and sort what is shown on his own *Alarms and Events* window independently of any other hpssgui user. See Section 3.10: *SSM List Preferences* on page 69 for information on filtering and sorting alarms. Filtering and sorting of the *Alarms and Events* list are not yet available for hpssadm users.

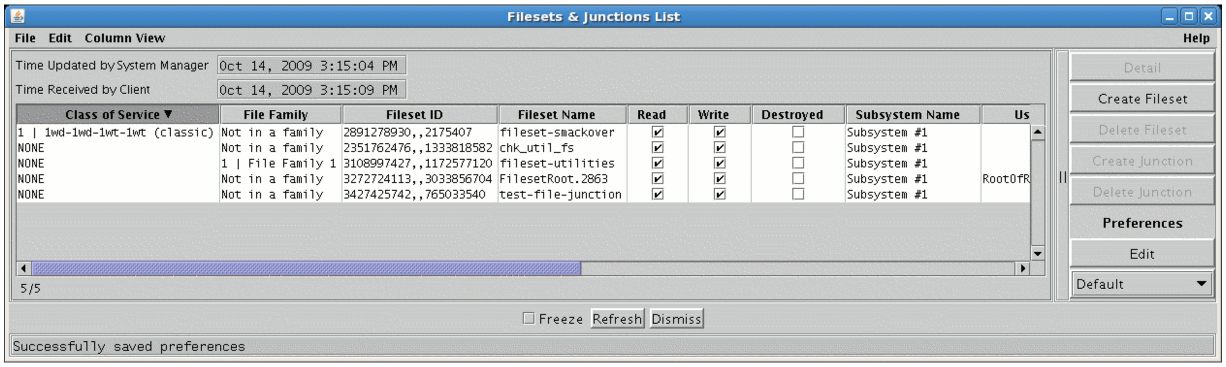
Chapter 10. Filesets and Junctions

A fileset is a logical collection of files that can be managed as a single administrative unit, or more simply, a disjoint directory tree. A fileset has two identifiers: a human readable name, and a numeric fileset ID. Both identifiers are unique to a given HPSS realm. Filesets are often used for HPSS name space administration. For example they can be used if a subtree of the HPSS name space needs to be assigned to a particular file family or class of service.

A junction is a Core Server object, much like a symbolic link, that is used to point to a fileset. This fileset may belong to the same Core Server or to a different Core Server. When pointing to a different Core Server, junctions allow HPSS users to traverse to different subsystems.

When a Core Server initializes for the first time on an empty database, it creates the 'root node' fileset. This is the fileset that contains the directory named '/'. For many sites this may be the only fileset needed and no other filesets will ever be created.

10.1. Filesets & Junctions List



Time Updated by System Manager: Oct 14, 2009 3:15:04 PM
Time Received by Client: Oct 14, 2009 3:15:09 PM

Class of Service ▼	File Family	Fileset ID	Fileset Name	Read	Write	Destroyed	Subsystem Name	Us
1 | 1wd-1wd-1wt-1wt (classic)	Not in a family	2891278930,,2175407	fileset-smackover	✓	✓	☐	Subsystem #1	
NONE	Not in a family	2351762476,,1333818582	chk_util_fs	✓	✓	☐	Subsystem #1	
NONE	1 | File Family 1	3108997427,,1172577120	fileset-utilities	✓	✓	☐	Subsystem #1	
NONE	Not in a family	3272724113,,3033856704	FilesetRoot.2863	✓	✓	☐	Subsystem #1	RootOfR
NONE	Not in a family	3427425742,,765033540	test-file-junction	✓	✓	☐	Subsystem #1	

5/5

Successfully saved preferences

This window displays information about the filesets and junctions that are configured in the HPSS system. From this window, detailed information about existing filesets and junctions can be viewed, new filesets and junctions can be created, and existing filesets and junctions can be deleted.

To open the window, from the *HPSS Health and Status* window Monitor menu select Filesets & Junctions.

There may be multiple entries in the list for a given fileset meaning that there are multiple junctions that point to the same fileset. Each entry for a given fileset will have a unique **Junction Name.** If the **Junction Name** is blank, then there are no junctions that point to the particular fileset.

The **lsfilesets**, **lsjunctions** and **lshpss** utilities may also be used to list all the filesets and junctions in the system. For more information, refer to the man pages for each utility.

Field Descriptions

Class of Service. The name of the Class of Service to which the fileset is assigned. If this field contains "NONE", the fileset has not been assigned to a Class of Service.

File Family. The name of the file family to which the fileset is assigned. If this field contains "Not in a family", the fileset has not been assigned to a family.

Fileset ID. The ID number which identifies the fileset. A fileset ID is displayed as two double-comma-separated unsigned integer numbers.

Fileset Name. The unique name which has been assigned to the fileset.

Read. If checked, the fileset is available for reading.

Write. If checked, the fileset is available for writing.

Destroyed. If checked, the fileset cannot be written to or updated.

Subsystem Name. The subsystem name to which the fileset has been assigned. This is the subsystem of the Core Server which controls the fileset.

User Data. This field is available for storing up to 128 bytes of data. The information can be ASCII or binary.

Directories. The number of directories in this fileset.

Files. The number of files in this fileset.

Hard Links. The number of hard links in this fileset.

Junctions. The number of junctions in this fileset.

Sym Links. The number of symbolic links in this fileset.

Junction Name. The name of a junction that points to this fileset. This field may be blank if there are no junctions configured to point to this fileset.

Parent Fileset. The name of the fileset in which the the junction resides. This field will be blank if the Junction Name is blank.

Buttons

Detail. Display the *Core Server Fileset Information Window* for the selected filesets. This button is only enabled if 1 or more items are selected in the list.

Create Fileset. Display the *Create Fileset Window* so that a new fileset may be created. This button is enable all the time.

Delete Fileset. Delete the selected filesets. This button is only enabled if 1 or more items are selected in the list.

Create Junction. Create a junction to the selected fileset. This button is only enabled when there is exactly 1 item selected in the list. The fileset name and id from the selected fileset will be used to fill in the information on the Create Junction Window. The user will only have to supply the Junction Name on the Create Junction Window to create the new Junction.

Delete Junction. Delete the selected junctions. This button is only enabled if 1 or more items are selected in the list.

10.2. Creating an HPSS Fileset

This section provides information on how to create HPSS filesets. Only the HPSS root user and SSM principal are allowed to create filesets.

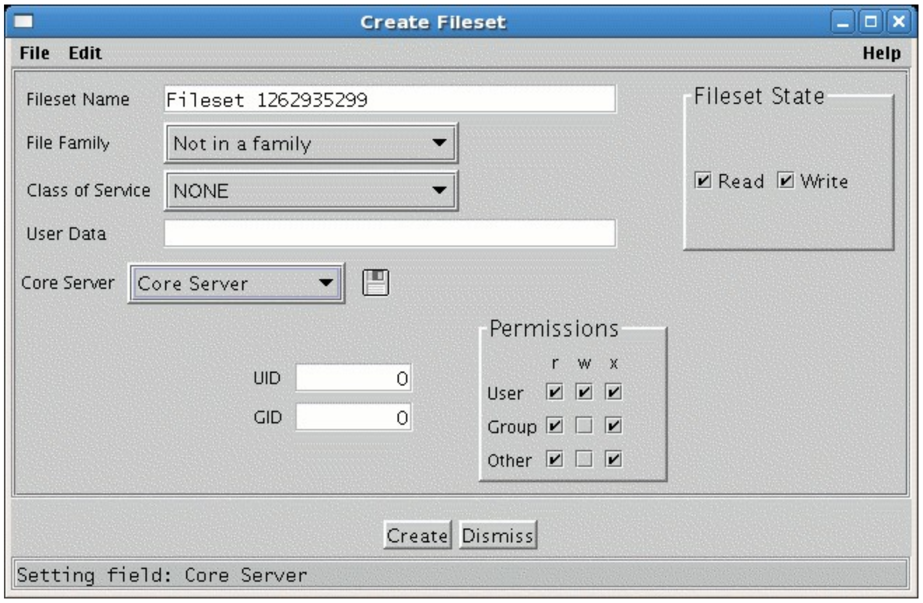
In order to successfully perform fileset administration, the DB2 Helper Program must be bound. See the HPSS Installation Guide, Section 5.8.1.3 Generate and Bind the DB2 Helper Program for more information.

An HPSS fileset can be created by using the Create Fileset SSM window, or by using the **create_fset** utility. *Refer to the* **create_fset** *man page for more information.*

SSM automatically generates a fileset ID for the new fileset.

After a fileset has successfully been created, a junction will need to be created to get to the new fileset name space from the root node fileset name space.

10.2.1. Create Fileset Window



Fileset Name: Fileset 1262935299
File Family: Not in a family
Class of Service: NONE
User Data:
Core Server: Core Server
UID: 0
GID: 0

Fileset State: ✓ Read ✓ Write

Permissions	r	w	x
User	✓	✓	✓
Group	✓		✓
Other	✓		✓

Setting field: Core Server

The *Create Fileset* window is used to create new HPSS filesets.

To open the window, from the *HPSS Health and Status* window Monitor menu select Filesets & Junctions to display the *Filesets & Junctions List,* then press the Create Fileset button.

After creating a new HPSS fileset, a new junction will be needed to connect this new fileset to the HPSS name space. See Section 10.5 *Creating a Junction* on page 315 for more information.

Field Descriptions

Fileset Name. The name to be assigned to the fileset. This name must be unique to the realm in which HPSS resides.

Fileset State. The state of the fileset. If Read is ON, the fileset will be available for reading. If Write is ON, the fileset will be available for writing.

File Family. The name of the File Family assigned to this fileset. If the File Family is to be other than the default, the File Family must have been previously created.

Class of Service. The name of the Class of Service assigned to this fileset. This menu will display the available Classes of Service.

Advice - File Family & Class of Service fields are optional. They provide the capability to force all data stored in the fileset to be assigned to a particular file family and/or class of service.

User Data. Any name or other data (up to 128 bytes) that the administrator wishes to associate with the fileset. The information can be ASCII or binary, although ASCII is easier to work with in this field. HPSS does not use this field in any way; it is strictly for user convenience in annotating the fileset. The field may be left blank.

The field is displayed as printable ASCII characters where possible. Non-printable bytes are displayed in backslash-octal notation, where each byte is shown as a backslash ("\") followed by a 3-digit octal number. For example, a tab character (decimal value 9) would be displayed as "\011". Backslash characters are themselves displayed as two backslashes ("\\"). Trailing null (i.e., zero) bytes are not displayed.

To modify this field, enter data in the same format. Printable characters (except for the backslash) can be entered normally. Backslash characters must be entered as "\\". Non-printable characters must be entered in backslash-octal notation. You need not specify leading zeros on the octal numbers, EXCEPT when the non- printable byte is followed by a printable octal digit character (0-7). In that case you must specify all 3 digits of the octal number to distinguish it from the printable character following.

Core Server. The name of the Core Server which will create and manage the fileset.

UID. The User ID identifying the user owning the root node of the fileset.

GID. The Group ID identifying the principal group owning the root node of the fileset.

Permissions. The initial UNIX-style permissions to be assigned to the root node of the fileset. There are nine checkboxes arranged in a matrix with the columns specifying "r" (read), "w" (write) and "x" (execute) permissions, and the rows specifying the three classes of users to which the permissions apply (User, Group, and Other). If a checkbox contains a check, it means that access is permitted. For example, if the checkbox in the "x" column and the Group row contains a check, it means that Group users have execute access.

Buttons

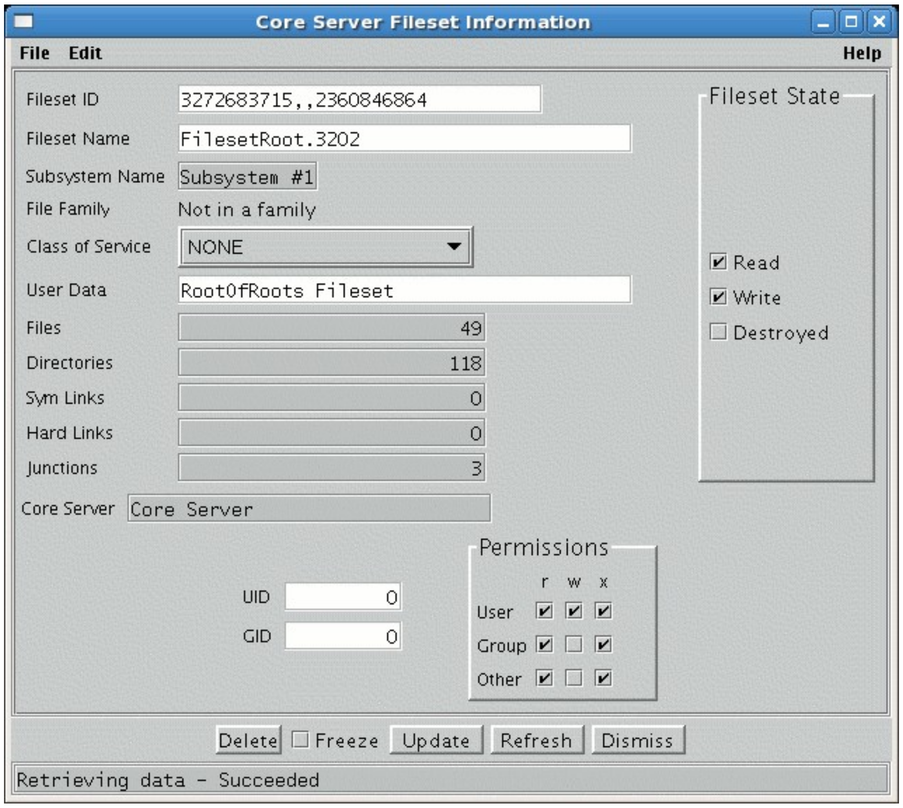
Create. Creates the fileset. If the create succeeds, a success message will appear on the status line at the bottom of the window; otherwise an error message will appear.

Dismiss. Closes the window.

10.3. Managing Existing Filesets

This section describes how to look up information on, modify, or delete filesets.

10.3.1. Core Server Fileset Information Window



Fileset ID: 3272683715,,2360846864
Fileset Name: FilesetRoot.3202
Subsystem Name: Subsystem #1
File Family: Not in a family
Class of Service: NONE
User Data: RootOfRoots Fileset
Files: 49
Directories: 118
Sym Links: 0
Hard Links: 0
Junctions: 3
Core Server: Core Server

Fileset State: [✓ Read] [✓ Write] Destroyed

UID: 0
GID: 0

Permissions

	r	w	x
User	✓	✓	✓
Group	✓		✓
Other	✓		✓

Retrieving data - Succeeded

This window allows an administrator to view, update and/or delete the Core Server information associated with a fileset. This information is acquired from the Core Server. While this window remains open, it will automatically update whenever change notifications about any field except UID, GID and permissions are received from the Core Server.

To open the window, from the *HPSS Health and Status* window Monitor menu select Filesets & Junctions to display the *Filesets & Junctions List,* select one or more filesets, then press the Detail button.

Any changes made to fields on this window are sent directly to the Core Server when the administrator clicks on the Update button; the changes are effective upon successful update. To mitigate automatic updates overwriting changes in progress, it would be wise to freeze this window by checking the Freeze checkbox before making changes.

The Fileset State can be used to prevent any changes to fileset data or to fileset metadata. Changing the state to just Read (unchecking the Write checkbox) will allow reading, but no changes will be allowed to the data or metadata. Changing the state to Destroyed will prevent both reading and writing.

Field Descriptions

Fileset ID. The ID number which identifies the fileset. A fileset ID is displayed as two double-comma-separated unsigned integer numbers. A new Fileset ID can be entered as two double-comma-separated unsigned integer numbers, as two single-period-separated unsigned integer numbers, as a 64-bit hexadecimal number that begins with '0x', or as an unsigned 64-bit number.

Fileset Name. The unique name which has been assigned to the fileset. The administrator can change the Fileset Name as long as the new name is unique to the realm in which HPSS resides

Subsystem Name. The subsystem name to which the fileset has been assigned. This is the subsystem of the Core Server which controls the fileset.

File Family. The name of the file family to which the fileset is assigned. If this field contains "Not in a family", the fileset has not been assigned to a family.

Class of Service. The name of the Class of Service to which the fileset is assigned. If this field contains "NONE", the fileset has not been assigned to a Class of Service.

Fileset State. This set of checkboxes displays fileset states, and allows them to be changed.

- Read - If checked, the fileset is available for reading.
- Write - If checked, the fileset is available for writing.
- Destroyed - If checked, the fileset cannot be written to or updated; in fact, it is not possible to set any of the attributes except the Fileset State attribute.

User Data. This field is available for storing up to 128 bytes of data. The information can be ASCII or binary, although ASCII is easier to work with in this window.

The field is displayed as printable ASCII characters where possible. Non-printable bytes are displayed in backslash-octal notation, where each byte is shown as a backslash ("\") followed by a 3-digit octal number. For example, a tab character (decimal value 9) would be displayed as "\011". Backslash characters are themselves displayed as two backslashes ("\\"). Trailing null (i.e., zero) bytes are not displayed.

To modify this field, enter data in the same format. Printable characters (except for the backslash) can be entered normally. Backslash characters must be entered as "\\". Non-printable characters must be entered in backslash-octal notation. It is not necessary to specify leading zeros on the octal numbers, EXCEPT when the non-printable byte is followed by a printable octal digit character (0-7). In that case, all 3 digits of the octal number must be specified.

Files. The number of files in this fileset.

Directories. The number of directories in this fileset.

Sym Links. The number of symbolic links in this fileset.

Hard Links. The number of hard links in this fileset.

Junctions. The number of junctions in this fileset.

Core Server. The name of the Core Server that handles this fileset.

UID. The User ID identifying the user owning the root node of the fileset.

GID. The Group ID identifying the principal group owning the root node of the fileset.

Permissions. The UNIX-style permissions assigned to the root node of the fileset. There are nine checkboxes arranged in a matrix with the columns specifying "r" (read), "w" (write) and "x" (execute) permissions, and the rows specifying the three classes of users to which the permissions apply (User, Group, and Other). If a checkbox contains a check, it means that access is permitted. For example, if the checkbox in the "x" column and the Group row contains a check, it means that Group users have execute access.

Buttons

Delete. Requests deletion of the currently displayed fileset. A confirmation will be requested before the request is sent to the Core Server. If the deletion is successful, a message will appear on the status line at the bottom of the window and all fields and buttons will be disabled (except for the Dismiss button). Note that only empty filesets may be deleted.

Freeze. Clicking on this checkbox freezes the window; in other words, all automatic updates to the window are suspended. Toggling the checkbox again returns the window, and the checkbox, to their normal conditions. Any accumulated changes that occurred while the window was frozen are immediately displayed.

Update. Requests that the currently displayed fileset information replace the fileset information currently kept by the Core Server.

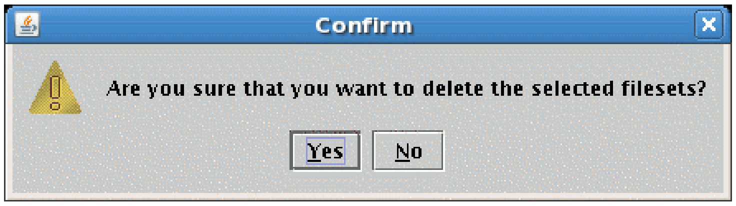
Refresh. Requests that the current information about this fileset be fetched from the Core Server.

Dismiss. Closes the window.

10.4. Deleting Filesets

Filesets must be empty in order to be deleted. The Core server will reject the deletion request if the fileset is not empty.

Filesets may deleted by selecting the filesets on the *Filesets & Junctions List* and pressing the Delete button. The following confirmation will be displayed;



Are you sure that you want to delete the selected filesets?

To continue with the fileset deletion, press “Yes”.

Filesets may also be deleted from the *Core Server Fileset Information* window using the Delete button.

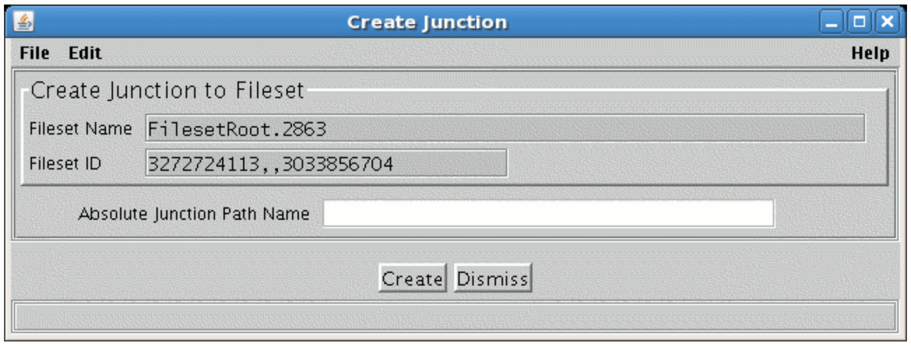
10.5. Creating a Junction

Only the HPSS root user and SSM principal are allowed to create junctions.

A junction is a name space object that points to a fileset and is similar to a persistent UNIX mount point. The fileset pointed to may reside in another subsystem.

Junctions can be created using SSM or by using the utility routine **crtjunction**. For more information, refer to the **crtjunction** man page.

10.5.1. Create Junction Window



Fileset Name FilesetRoot.2863
Fileset ID 3272724113,,3033856704
Absolute Junction Path Name

This window allows you to create a junction named by the Absolute Junction Path Name to the specified fileset. HPSS defines a junction as a name space object that 'points' to a directory. A fileset is an independent directory subtree, managed as a unit. The fileset to which the junction will point may be managed by the same Core Server where the junction is being created or it may be managed by another HPSS Core Server. HPSS junctions are similar to persistent UNIX mount points.

Although HPSS allows junctions to any directory, this window can only be used to create junctions to directories that are also filesets.

Field Descriptions

Fileset ID. The ID number which identifies the fileset to which the junction will point. This field is filled in from the Fileset ID selected on the *Filesets & Junctions List*.

Fileset Name. The unique name which identifies the fileset to which the junction will point. This field is filled in from the Fileset Name selected on the *Filesets & Junctions List*.

Absolute Junction Path Name. The absolute path name, in the HPSS name space, of the junction that is to be created. If the path name to the junction is not valid, the creation will fail.

Buttons

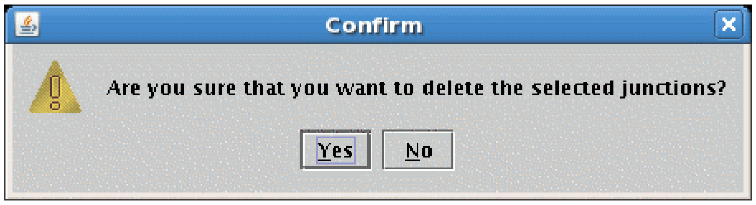
Create. Creates the junction. If the create succeeds, a success message will appear on the status line at the bottom of the window; otherwise an error message will be displayed. At this point, data can be entered for another junction, or the window can be dismissed.

Dismiss. Closes the window.

10.6. Deleting a Junction

Junctions can be deleted using SSM or by using the utility routine **deljunction**. For more information, refer to the **deljunction** man page.

To delete a junction using SSM, select the junction(s) to be deleted from the *Filesets & Junctions List* and press the Delete Junction button. You will be asked to confirm the deletion with the following dialog;



Are you sure that you want to delete the selected junctions?

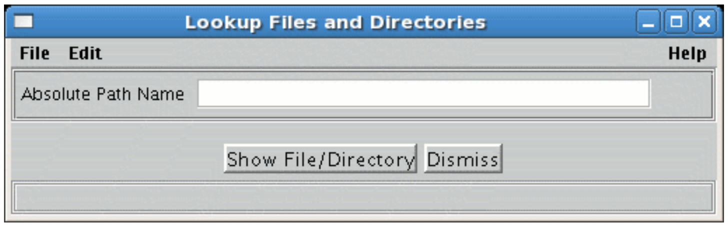
To continue with the junction deletion, press “Yes”.

Chapter 11. Files, Directories and Objects by SOID

This chapter describes two ways to display basic information about files and directories stored in HPSS. Starting with a fully qualified path name, you can look up a file or directory in the system and display information about it. In other cases, your starting point may be a SOID, a Storage Object ID, which is the internal computer generated name for files, directories, and virtual volumes. Some utility programs report the names of objects of interest as SOIDs.

SOIDs are more difficult to deal with manually as they are not intended to to be used that way. But, they contain all the necessary information to identify the server, object, and object type, so they can be used to zero-in on a very specific internal data structure, such as a bitfile or virtual volume, that would otherwise be very difficult to isolate and display.

11.1. Files & Directories Window



This window is reached through **Monitor – Lookup HPSS Objects – Files & Directories**. The pathname of either an HPSS file or directory may be entered. Use the Show File/Directory button to display specific information.

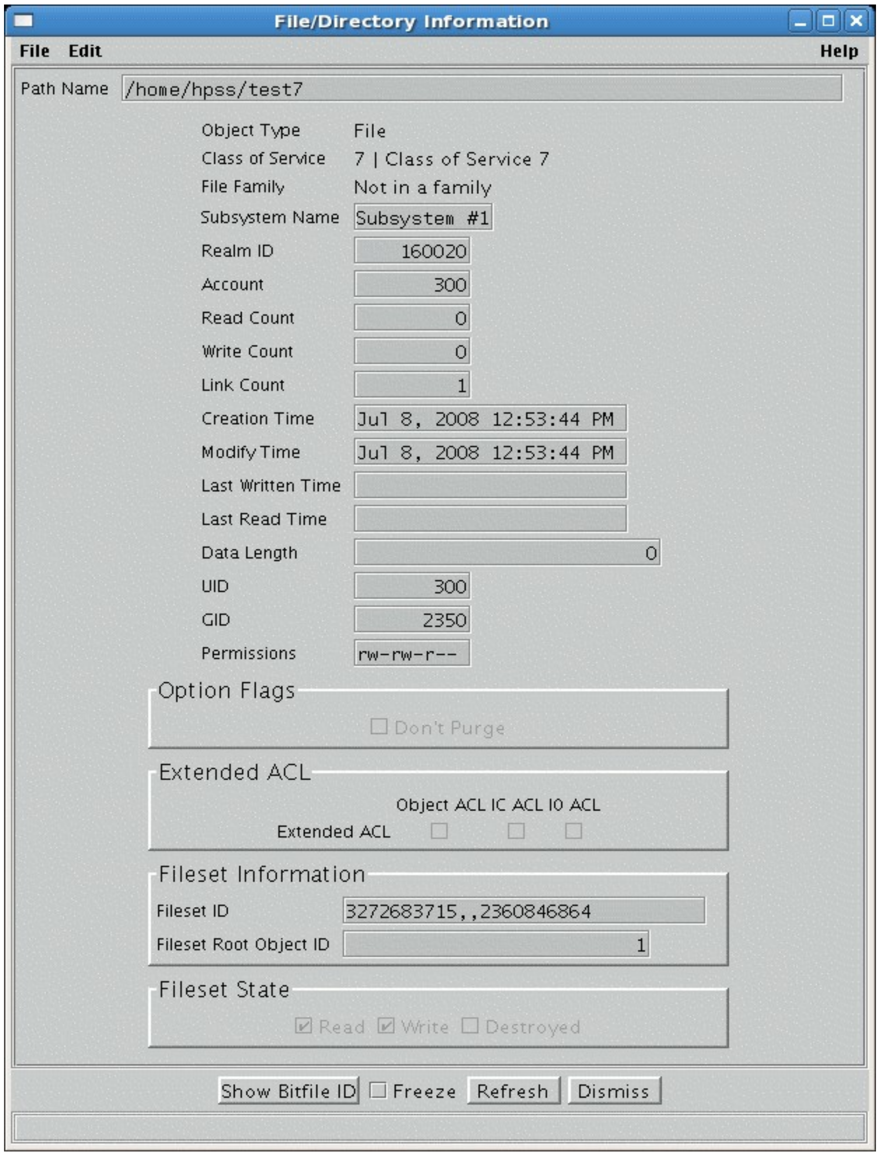
Field Descriptions

Absolute Path Name. The absolute path name (or fully qualified path name), in the HPSS name space, of the file or directory that is to be displayed. If the path name to the file or directory is not valid, then lookup will fail. If the path name is to a symbolic link or junction, then the object displayed will be the object pointed to by the symbolic link or junction.

Show File/Directory. If the Absolute Path Name exists, clicking on this button will open the *File/Directory Information* window. The *Lookup Files and Directories* window will remain open so that the administrator may specify another file or directory.

Dismiss. Closes the window.

11.1.1. File/Directory Information Window



Path Name	/home/hpss/test7
Object Type	File
Class of Service	7 | Class of Service 7
File Family	Not in a family
Subsystem Name	Subsystem #1
Realm ID	160020
Account	300
Read Count	0
Write Count	0
Link Count	1
Creation Time	Jul 8, 2008 12:53:44 PM
Modify Time	Jul 8, 2008 12:53:44 PM
Last Written Time	
Last Read Time	
Data Length	0
UID	300
GID	2350
Permissions	rw-rw-r--

Option Flags

☐ Don't Purge

Extended ACL

	Object ACL	IC ACL	IO ACL
Extended ACL	☐	☐	☐

Fileset Information

Fileset ID	3272683715,,2360846864
Fileset Root Object ID	1

Fileset State

☑ Read ☑ Write ☐ Destroyed

This window shows details about a file or directory.

If a file is displayed in the *File/Directory Information* window, a button labeled Show Bitfile ID will appear at the bottom of the window. Pressing this button will cause the *Storage Object ID* window to appear.

Field Descriptions

Path Name. The pathname to either the file or the directory.

Object Type. The type of object being displayed, either File or Directory.

Class of Service. The name of the Class of Service in which the file is stored. If the displayed object is a directory the value of this field will be NONE.

File Family. The name of the file family to which the file has been assigned. If the file has not been assigned to a family, the value of this field will be "Not in a family".

Subsystem Name. The name of the HPSS subsystem which contains the file or directory.

Realm ID. The ID number which identifies the realm which encompasses the file or directory.

Account. The account index that is associated with the file or directory. This value is used by the HPSS accounting sub-system.

Read Count. This field applies only to files and is the number of read operations that have been issued against the file. Reading a file through most HPSS interfaces may cause multiple read operations to be issued, so this value may increase by more than one with each user read of a file.

Write Count. This field applies only to files and is the number of write operations that have been issued against the file. Writing a file through most HPSS interfaces may cause multiple write operations to be issued, so this value may increase by more than one with each user write of a file.

Link Count. For directories, this is the count of the number of directories that point to this directory. This includes Dot and DotDot.

Creation Time. Time and date that the file or directory was created.

Modify Time. Time and date that the metadata associated with the file or directory was last modified.

Last Written Time. This field only applies to files and is the time and date of the last write to the file. If this field is blank, the file has never been written.

Last Read Time. Time and date of the last read from the file, or the time and date of the last read from the directory.

Data Length. For directories, this is the byte length of the metadata associated with this directory. For files, this is the largest numbered byte in the file. Note that this is not necessarily the number of bytes in the file as files can be written sparsely with gaps. Also, it is not necessarily the highest written byte, since the Core Server supports POSIX clear and truncate commands which can leave a gap at the end of the file.

UID. The user ID of the owner of the file or directory.

GID. The group ID of the owner of the file or directory.

Permissions. Permissions granted to the file or directory. These are displayed in the standard UNIX mode format (read, write, execute permissions for user, group, and other). These are base permissions and do not take into account the effects of any ACLs on the file or directory.

Option Flags

> **Don't Purge**. This field applies only to files. If the **Don't Purge** check box is selected the file is purge-locked and will not be purged from the top level of the hierarchy. If the **Don't Purge** check box is not selected the file may be purged as usual.

Extended ACL. If any ACL entry other than the default ACL entries exist, then the file or directory is said to contain extended ACLs. There are three type of ACLs that could have extended ACLs:

- Object ACL – HPSS Name Space Object ACL
- IC ACL – HPSS Name Space Initial Container ACL
- IO ACL – HPSS Name Space Initial Object ACL

A check mark will be put into each of the ACLs containing extended ACL entries for this file or directory. For example, if a directory contains an USER ACL entry in the Object ACL and Initial Container ACL, then both the Object ACL and IC ACL checkboxes will contain check marks.

Fileset Information

The Fileset Information group contains key fileset attributes for the root node of the fileset containing the file or directory.

> **Fileset ID**. The ID number which identifies the fileset containing the file or directory . The 64-bit Fileset ID is displayed as two 32-bit unsigned integers separated by two commas.
>
> **Fileset Root Object ID**. The ID number which identifies the root node of the fileset which contains the file or directory. This value is a 64-bit object ID.

Fileset State. Filesets have three access states - read, write, and destroyed. If a fileset allows reading, a check mark is displayed in the box labeled Read. If a fileset allows data to be written, a check mark is displayed in the box labeled Write. If a fileset is turned off, a check mark is displayed in the box labeled Destroyed.

Show Bitfile ID. This button is displayed only for file objects. It opens a separate window to display the Storage Object ID which identifies the Core Server bitfile.

Freeze. Clicking on this checkbox freezes the window; in other words, all automatic updates to the window are suspended. Toggling the checkbox again returns the window, and the checkbox, to their normal conditions. Any accumulated changes that occurred while the window was frozen are immediately displayed.

Refresh. Requests that the current information about this fileset be fetched from the Core Server.

Dismiss. Closes the window.

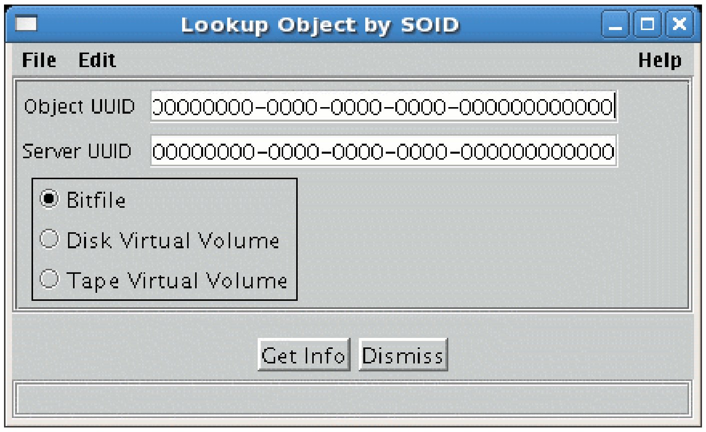
Note: The table below indicates which fields/attributes apply to both files and directories or just one of the two. Please refer to it to see which fields on the *File/Directory Information* window apply to files and/or directories.

Attributes of Files/Directories

Attribute	**File**	**Dir**
Account	x	x
BitfileId	x	
Attribute	**File**	**Dir**
RealmId	x	x

Comment	x	x
CompositePerms	x	x
COSId	x	
DataLength	x	x
EntryCount		x
ExtendedACLs	x	x
FamilyId	x	
FilesetHandle		
FilesetId	x	x
FilesetRootId	x	x
FilesetStateFlags	x	x
FilesetType	x	x
GID	x	x
GroupPerms	x	x
LinkCount		x
ModePerms	x	x
Name	x	x
OpenCount	x	
OptionFlags	x	
OtherPerms	x	x
ReadCount	x	
SubSystemId	x	x
TimeCreated	x	x
TimeLastRead	x	x
TimeLastWritten	x	x
TimeModified	x	x
Type	x	x
UID	x	x
UserPerms	x	x
WriteCount	x	

11.2. Objects by SOID Window



Object UUID 00000000-0000-0000-0000-000000000000

Server UUID 00000000-0000-0000-0000-000000000000

● Bitfile
○ Disk Virtual Volume
○ Tape Virtual Volume

To display this window, select **Monitor** from the *Health and Status* window, then select **Lookup HPSS Objects,** and then"**Objects by SOID**. This window allows you to open an information window for an HPSS object which you specify by the object's HPSS Storage Object ID (SOID). The object types supported by this screen are Bitfiles and Virtual Volumes (Disk and Tape).

Field Descriptions

Object UUID. The UUID of the object. This must be entered in the indicated format. HPSS utility programs that display SOIDs should present this field in the correct format.

Server UUID. The UUID of the server which manages the object. This must be entered in the indicated format. HPSS utility programs that display SOIDs should present this field in the correct format.

Type

This is a collection of radio buttons which help to specify the type of storage object which this SOID identifies.

Bitfile. A bitfile lookup.

Disk Virtual Volume. A lookup for a disk virtual volume will be performed.

Tape Virtual Volume. A lookup for a tape virtual volume will be performed.

Buttons

Get Info. Once all of the fields have been filled in, clicking on this button will open the information window for the object type you selected. The *Lookup Object by SOID* window will remain open so that another SOID may be specified.

Dismiss. Closes the window.

Chapter 12. Tape Aggregation

This chapter discusses the following operations:

- Overview of Tape Aggregation
- Tape Aggregation Performance Considerations
- Configuring Tape Aggregation

12.1. Overview of Tape Aggregation

Tape aggregation reduces file processing time when migrating relatively small files from disk to tape. In certain situations it may improve very small file disk-to-tape migration performance by two orders of magnitude over normal migration.

During normal migration, HPSS migrates each file into a separate tape segment. This generally causes the tape drive to flush the file data to the tape, and write a tape mark. It is not unusual for the majority of the time spent migrating small files to be dominated by this.

If tape aggregation is enabled in the Disk Migration Policy, and if there are a sufficient number of migration candidate files ready to be migrated, HPSS will pack the data of many small files into a single tape storage segment and only write a single tape mark for the entire "batch".

Note that this is managed internally by HPSS and is transparent to users. For example, it has no effect on file security or how files are accessed. It generally has little effect on file retrieval performance.

12.2. Tape Aggregation Performance Considerations

In order to get good performance with tape aggregation, there are several things to consider.

First, there need to be a sufficient number of files ready to migrate. If migration runs too often, or if the HPSS system doesn't tend to ingest a large number of small files in a relatively short period of time, then there may be little or no benefit to tape aggregation since there will be very few files, if any, in each aggregate.

Second, tape aggregation performs best with relatively small files since most of the time spent to migrate them occurs during tape mark processing when the data is flushed to the media. It usually isn't useful to aggregate large files since the tape mark processing time is dominated by the time needed to write the file data.

Finally, tape aggregates are currently written as a unit. If there is a problem writing a very large aggregate (e.g. end of media is reached) the entire aggregate will need to be rewritten again later. If the aggregate is very large (e.g. 10% of a large capacity tape), this may take a long time. In general it is more important to set up your Disk Migration Policy so that a large majority of the total number of files you expect to receive are put into aggregates, rather than being overly concerned with the amount of data being put into aggregates.

12.3. Configuring Tape Aggregation

To turn on tape aggregation, make sure the Aggregate Files to Tape option is enabled on the Disk

Migration Policy screen. Edit any other tape aggregation related fields on that screen as needed. If MPS is running, you must also tell it to reread the Disk Migration Policy.

Chapter 13. User Accounts and Accounting

13.1. Managing HPSS Users

After the HPSS system is up and running, the administrator must create the necessary accounts for the HPSS users. For a new HPSS user, a Kerberos, LDAP, or UNIX ID (depending on authentication type configured) and an FTP ID must exist before the user can access HPSS via FTP. In addition, if the HPSS user needs to use SSM, an SSM ID must also be created before the user can use SSM. The SSM ID should be created only for the HPSS administrators and operators.

The HPSS User Management Utility (hpssuser) provided with HPSS can be used by the administrator to add, delete, and list the HPSS user IDs. The utility must run as root to acquire the necessary authority to create new KRB, LDAP, UNIX, FTP, and SSM IDs. Refer to the hpssuser man page for more information on how to invoke the hpssuser utility.

13.1.1. Adding HPSS Users

The hpssuser utility can be used by the administrator to add a UNIX User ID, a KRB User ID, an LDAP User ID, an FTP User ID, and an SSM User ID to the HPSS if these IDs do not already exist. The hpssuser utility can be invoked to simultaneously add multiple types of User IDs for a user, or to add an individual user type. Refer to the hpssuser man page for more information.

Ensure that the Core Server is up and running before adding the FTP User ID. The hpssuser utility will not be able to create the user's home directory if the Core Server is not available.

13.1.1.1. Add All User ID Types

The utility can be used to simultaneously add all relevant User ID types for a user. When invoked with the '-all' option, the hpssuser utility will consult the system authentication and authorization configuration and add user types which are consistent with the configuration. For example, if the system is configured to use Kerberos authentication and LDAP authorization, a KRB User ID and an LDAP User ID will be added in addition to the UNIX User ID, FTP User ID and SSM User ID types.

Invoke the **hpssuser** utility as follows to add the required User ID for an HPSS user:

```
hpssuser -add <user> -all
```

When invoked, the **hpssuser** utility will prompt the user for any required data. The following is an example of adding all User ID types on a system configured to use Kerberos authentication and LDAP authorization. The **-nohome** option indicates that no FTP home directory will be created.

```
# hpssuser -add user1 -all -nohome
User ID#: 300
Primary group name: hpss
Enter password for user1: ******
Re-enter password to verify: ******
Full name: Test User
Login shell: /bin/ksh
Unix (local/system) home directory: /home/user1
[ adding unix user ]
[ added unix user ]
[ KADMIN_PRINC unset; using kadmin.local for Kerberos ops ]
[ adding kerberos principal ]
[ added kerberos principal ]
HPSS/LDAP home directory: /
Primary group ID#: 210
[ adding ftp user ]
[ ftp user added ]
[ adding ldap principal ]
[ added ldap principal ]
[ adding ssm user ]
1) admin
2) operator
Choose SSM security level
(type a number or RETURN to cancel):
> 1
[ ssm user added : admin ]
```

If the -nohome option is not specified when adding an FTP user, you must authenticate (using kinit) as that user before running hpssuser, or hpssuser will be unable to create the FTP home directory for the user. Because of this limitation, you MUST supply the -nohome option when adding a user using the -all flag.

13.1.1.2. Add a UNIX User ID

The **hpssuser** utility can be used to create UNIX identities and to create unix keytabs by using the following syntax:

```
hpssuser -add <user> -unix [-unixkeytab <path>]
```

The utility will prompt the user for required data. Following is an example of adding a UNIX User ID and creating a UNIX keytab:

```
# hpssuser -add user1 -unix -unixkeytab user1.keytab
User ID#: 300
Primary group name: hpss
Enter password for user1: ******
Re-enter password to verify: ******
Full name: Test User
Login shell: /bin/ksh
Unix (local/system) home directory: /home/user1
[ adding unix user ]
[ added unix user ]
[ adding unix keytab entry to 'user1.keytab' ]
[ added unix keytab entry to 'user1.keytab' ]
```

13.1.1.3. Add a Kerberos User ID

The **hpssuser** utility invokes the kadmin utility to create the KRB principal and account. This can be done using both keytab and password by specifying the -krbkeytab option.

Invoke the **hpssuser** utility as follows to add a KRB User ID using a keytab and password:

```
hpssuser -add <user> -krb [-krbkeytab <path>]
```

The utility will prompt the user for the required data. Following is an example of adding a KRB User ID using keytab and password:

```
# hpssuser -add user1 -krb -krbkeytab user1.krb.keytab
[ adding kerberos principal ]
[ KADMIN_PRINC unset; using kadmin.local for Kerberos ops ]
Enter Kerberos password for user1: ******
Re-enter password to verify: ******
[ adding kerberos keytab entry to 'user1.krb.keytab' ]
[ added kerberos keytab entry to 'user1.krb.keytab' ]
[ added kerberos principal ]
```

13.1.1.4. Add an LDAP User ID

Invoke the **hpssuser** utility as follows to add an LDAP User ID:

```
hpssuser -add <user> -ldap
```

The utility will prompt the user for the required data. Following is an example of adding an LDAP User ID:

```
# hpssuser -add user1 -ldap
[ adding ldap principal ]
User ID#: 300
HPSS home directory: /home/user1
Primary group ID# 210
Enter LDAP password: ******
Re-enter password to verify: ******
[ ldap user added ]
```

13.1.1.5. Add an FTP User ID

The **hpssuser** utility adds a password entry in the FTP Password file and creates the user's home directory in HPSS. Note that if FTP is configured to use the UNIX password file, then a UNIX user will be added.

Invoke the **hpssuser** utility as follows to create an FTP User ID:

```
hpssuser -add <user> -ftp
```

The utility will prompt the user for the required data. Following is an example of adding an FTP User ID on a system configured to use a separate password file FTP than for UNIX:

```
# hpssuser -add user1 -ftp -nohome
User ID#: 300
Enter password for user1: ******
Re-enter password to verify: ******
Full name: Test User
HPSS/LDAP home directory: /home/user1
Login shell: /bin/ksh
Primary group ID#: 210
[ adding ftp user ]
[ ftp user added ]
```

If the -nohome option is not specified when adding an FTP user, you must authenticate (using kinit) as that user before running hpssuser.

Ensure that the Core Server is up and running before adding the FTP User ID. The hpssuser utility will not be able to create the user's home directory if the Core Server is not available.

13.1.1.6. Add an SSM User ID

The hpssuser utility creates an SSM ID for a user by adding an entry for him to the AUTHZACL table. Refer to Section 2.3: *SSM User Security* on page 26 for more information on SSM user security.

Invoke the hpssuser utility as follows to add an SSM User ID:

```
hpssuser -add <user> -ssm
```

The utility will prompt the user for the required data. Following is an example of adding an SSM User ID:

```
# hpssuser -add user1 -ssm
[ adding ssm user ]
1) admin
2) operator
Choose SSM security level
(type a number or RETURN to cancel):
> 1
[ ssm user added : admin ]
```

13.1.2. Deleting HPSS Users

The **hpssuser** utility can be used by the administrator to delete existing User IDs for an HPSS user. The utility can be invoked to delete all User IDs for the user or to delete an individual ID.

The utility will prompt the user for the required data. Following is an example of deleting the User IDs for an HPSS user on a system configured to use Kerberos authentication and LDAP authorization:

```
# hpssuser -del user1 -all
[ deleting ssm user ]
[ SSM user deleted ]
[ deleting ldap principal ]
[ deleted ldap principal ]
[ deleting ftp user ]
[ ftp user deleted ]
[ deleting kerberos principal ]
[ KADMIN_PRINC unset; using kadmin.local for Kerberos ops ]
[ deleted kerberos principal ]
[ deleting unix user ]
```

13.1.3. Listing HPSS Users

The hpssuser utility can be used by the administrator to list all existing HPSS User IDs. The utility can be invoked to list all HPSS User IDs or a particular type of User ID.

Following is an example of listing the user1 User ID on a system configured to use UNIX authentication and authorization, and with FTP configured to use the UNIX password file:

```
# hpssuser -list user1 -all
[ Kerberos not configured - skipping... ]
INFO: FTP is configured to use the unix password file.
[ LDAP not configured - skipping... ]
[ unix(linux) user info ]
uid=300
gid=210
home=/home/user1
shell=/bin/ksh
fullname=Test User
[ ssm user info ]
  (admin    -> perms = 'rwxcidt')
  (operator -> perms = 'r--c--t')
SSM client security ACL entry:
rwxcidt - user - 300 (user1) - 10000 (HPSS_REALM.NAME)
```

Listing all unix users is not supported.

13.1.4. Create an SSM Client Package

The **hpssuser** utility can be used by the administrator to generate an SSM Client Package (all files required to start an SSM client on another node). The utility also has the ability to include HPSS documentation in the package if the -packagedoc option is specified. The following is an example of creating a SSM Client Package without the HPSS documentation on a system configured to use Kerberos authentication. For the case of a system configured to use UNIX, then krb5.conf won't be included in the package. Note that if krb5.conf does not exist on a system that's configured to use krb auth, then the command will fail.

```
# hpssuser -ssmclientpkg /tmp/ssmclientpkg.tar
[ packaging ssm client ]
[ creating /tmp/ssmclientpkg.tar ]
ssm.conf
krb5.conf
hpssgui.pl
hpssgui.vbs
hpss.jar
[ packaged ssm client in /tmp/ssmclientpkg.tar ]
```

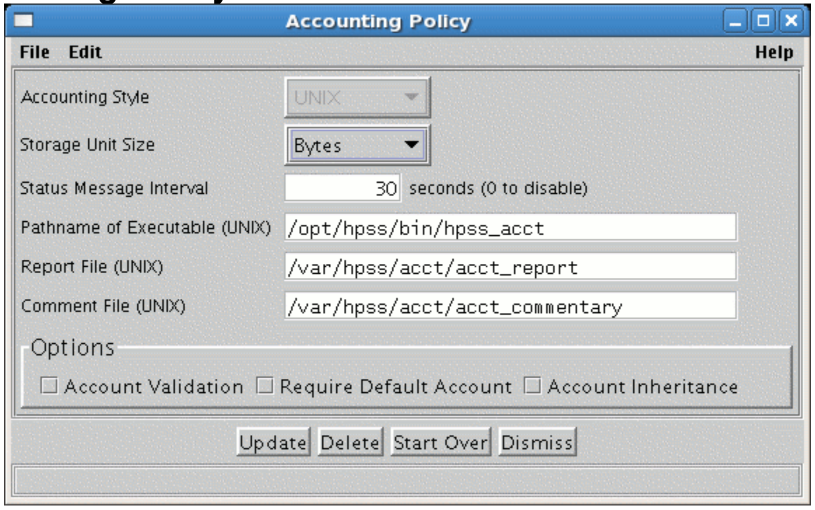
13.2. Accounting

HPSS maintains accounting information on the usage of the system whether the site charges for usage or not. Sites are encouraged to use the accounting information, even if they do not charge users, to gain a better understanding of the usage patterns of their system.

The accounting policy defines how the usage of HPSS resources is reported. An accounting policy is required whether the site actually charges users for HPSS usage or not.

The accounting policy may be created using the *Accounting Policy* window (select this window by clicking **Accounting** on the **Policies** submenu of the **Configure** menu on the *HPSS Health and Status* window). After the accounting policy is created, it can be viewed, updated, or deleted through the same window. Note, however, that once the Accounting Style is chosen, it may not be modified.

13.2.1. Accounting Policy Window



Accounting Style	UNIX
Storage Unit Size	Bytes
Status Message Interval	30 seconds (0 to disable)
Pathname of Executable (UNIX)	/opt/hpss/bin/hpss_acct
Report File (UNIX)	/var/hpss/acct/acct_report
Comment File (UNIX)	/var/hpss/acct/acct_commentary

☐ Account Validation ☐ Require Default Account ☐ Account Inheritance

This window allows an administrator to manage the accounting policy. If the Update button is visible, then the accounting policy already exists; otherwise an Add button will be displayed. If any option is updated on the window, the Core Server and Gatekeeper will need to be recycled before changes will take effect.

An accounting policy is required whether the site actually charges users for HPSS usage or not.

Field Descriptions

Accounting Style. The style of accounting that is used by the entire HPSS system. Valid values are SITE or UNIX. The default value is UNIX.

Under UNIX style accounting, resource usage is reported by user ID (UID). Each user is allowed to use only one account ID, which has the same numerical value as his user ID. Under SITE style accounting, each user may user multiple account IDs.

Note that if you use SITE style accounting, you must either enable **Account Validation** in the Accounting Policy or select LDAP as your authorization mechanism. See below for a description of the **Account Validation** field in the Accounting Policy. See section 2.1:Security Services on page 21for a description of LDAP and other HPSS authorization mechanisms. See the *HPSS Installation Guide*, section 3.9.3: *Accounting Policy and Validation* for a complete description of accounting styles and account validation.

Advice - This field must be set up properly before any files are written to the system. Under UNIX-style accounting, HPSS obtains the account number from the user's UNIX UID. Site-style accounting allows the account ID to be set independently of the UNIX UID. Once the accounting policy is configured, the Accounting Style cannot be changed.

Storage Unit Size. The integral units to be displayed in the report file. The valid values for this field are Bytes, Kilobytes, Megabytes, and Gigabytes. The default value is Bytes.

Status Message Interval. The number of seconds between status messages sent to SSM by the Accounting utility during an accounting run. A non-zero value less than 10 will be treated as 10. A value of 0 will prevent accounting status messages from being generated. The default value is 30 seconds.

Pathname of Executable (UNIX). The UNIX path name of the accounting utility executable. This field may contain any valid UNIX path name. The default value is "**/opt/hpss/bin/hpss_acct**".

Report File (UNIX). The UNIX pathname where the generated accounting report will be stored. The Report File (UNIX) field can contain any valid UNIX pathname. The default value is "**/var/hpss/acct/acct_report**".

Comment File (UNIX). A UNIX pathname of an optional commentary text file. The contents of this file, if it exists, are prepended to the Report File (UNIX) at the end of a successful accounting run. The data field may be left empty to indicate that a comment file will not be used or it may contain any legal UNIX pathname. The default value is "**/var/hpss/acct/acct _commentary**".

Options:

Account Validation. A flag that indicates whether or not authorization of user accounts should be performed by a Gatekeeper. The default value is OFF.

Advice - If you turn this flag ON, you must configure at least one Gatekeeper which will perform the account validation service.

Require Default Account. A flag that indicates whether or not users must have a valid default account index before they are allowed to perform any operation. The default value is OFF. It is only used if Account Validation has been enabled. If this flag is disabled, validation will occur only during file manipulation operations.

Account Inheritance. A flag that indicates whether or not newly created files and directories should automatically inherit the account index used by their parent directory. The default value is OFF. It is only used if Account Validation has been enabled and Site- style accounting has been selected. If this flag is disabled, new files and directories have the user's current session account index applied to them.

13.2.2. Accounting Reports and Status

13.2.2.1. Generating an Accounting Report

The HPSS Accounting Report tool accumulates accounting information from metadata and generates a report file. The accounting report will include:

- Total file accesses, total number of files, and total space used per account per class of service.
- Total file accesses and amount of data transferred per account per class of service per storage class.

The accounting report represents a blurred snapshot of the system storage usage as it existed during the accounting run. By "blurred snapshot" we mean that the accounting information is gathered while the system runs, so the accounting data is changing underneath it, and it may not represent what would be collected if no users were active during the accounting process. Accounting information is maintained per storage subsystem and the accounting reports summarize information for that storage subsystem.

Before creating an accounting report, you must create and fully configure an Accounting Policy. The initial Accounting Policy should be set up before any files are created in the HPSS system. Except for the Accounting Style, the Accounting Policy fields can be updated any time after the initial setup.

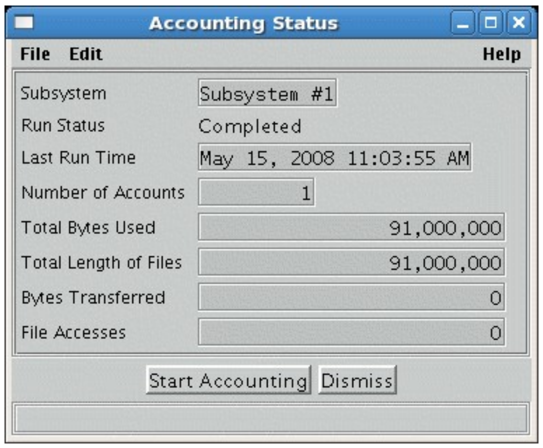
There are two ways to generate an accounting report: from the *Subsystems* window or from the *Accounting Status* window. For each method, begin by selecting the Accounting Report menu item from the Operations menu on the *HPSS Health and Status* window. This will bring up the *Subsystems* window. Click on the storage subsystem for which an accounting report will be generated. This will highlight the Accounting Status and Start Accounting buttons.

To create an accounting report, choose either of these methods:

- On the *Subsystems* window, click the Start Accounting button. The accounting report should run quickly and a completion status will appear at the bottom of the *Subsystems* window.
- Bring up the *Accounting Status* window by clicking on the Accounting Status button on the *Subsystems* window. On the *Accounting Status* window, click the Start Accounting button. The accounting report should run quickly and a completion status will appear at the bottom of the *Accounting Status* window.

For either method, the *Accounting Status* window will display the overall statistics from the last accounting run.

13.2.2.2. Accounting Status Window



Subsystem	Subsystem #1
Run Status	Completed
Last Run Time	May 15, 2008 11:03:55 AM
Number of Accounts	1
Total Bytes Used	91,000,000
Total Length of Files	91,000,000
Bytes Transferred	0
File Accesses	0

This window allows an administrator to view the accounting status and start accounting.

Field Descriptions

Subsystem. The name of the storage subsystem containing this accounting status data.

Run Status. Current status of accounting run. Possible values are:

- Never run
- Running
- Failed
- Completed
- Report generated

Last Run Time. If accounting is currently running, this is the time the run started. Otherwise it is the time the last run completed.

Number of Accounts. Total number of accounts in the storage subsystem. Set after a successful run.

Total Bytes Used. Total number of bytes accounted for in the storage subsystem. Set after a successful run.

Total Length of Files. Total length of all bitfiles in the storage subsystem. Set after a successful run.

Bytes Transferred. Total bytes transferred for the storage subsystem during the accounting period covered by the last successful accounting run.

File Accesses. Total file accesses for the storage subsystem during the accounting period covered by the last successful accounting run.

Start Accounting. Causes the accounting utility to start running for the storage subsystem. As it runs, the status fields will be updated after the statistics have been changed by the accounting utility.

13.2.2.3. Interpreting the Accounting Report

The default accounting output program for HPSS generates a text file that contains two types of lines.

The first type, denoted by a zero (0) in the fourth column, gives the following summary information about the storage used by a particular HPSS Account Index (AcctId) in a particular Class Of Service (COS):

- The total number of file accesses (#Accesses) to files owned by the Account Index in the Class Of Service. In general, file accesses are counted against the account of the user accessing the file, not the owner of the file itself.
- The total number of files (#Files) stored under the Account Index in the Class Of Service.
- The total amount of data stored (Length) under the Account Index in the Class Of Service.

The second type of line has a non-zero value in the Storage Class column (Sclass). This type of line contains information about the storage used by a particular HPSS Account Index (AcctId) in a particular Storage Class (Sclass) within a particular Class Of Service (COS). These lines contain the following information:

- The total number of file accesses (#Accesses) in this particular Storage Class for the given Class Of Service. If a class of service is configured to stage bitfiles on open, then all file accesses will occur in the storage class at the top of the hierarchy.
- The total amount of data transferred (Transferred) into or out of this Storage Class and Class Of Service for this particular Account Index. Note that data transferred is counted against the owner of the account transferring the data, not the owner of the data itself.

Note: File transfer information is not always accounted for when it occurs through interfaces other than the HPSS Client API.

Example Accounting Report File:

```
# Comment file first line
# Comment file last line
# HPSS Accounting Snapshot completed on Wed Jul 15 12:57:00 1998
# for Subsystem 1
# Storage Unit Size : 1
# Total Number of Rows : 6
# Total Number of Accounts : 2
# Total Storage Units for HPSS system : 15598533
#
# Entries with ‘0’ in the SClass field are COS totals.
# Other entries apply to individual storage classes.
#
# Realmid  AcctId     COS  0       #Accesses  #Files      Length(COS)
# Realmid  AcctId     COS  Sclass  #Accesses  Transferred    (SClass)
# -------------------------------------------------------------------
  0        2033       3    0          0         5          212895
  0        634        1    0          89        125        4168147
  0        634        1    1          87        201256
  0        634        1    2          2         0
  0        634        5    0          152       152        11217491
  0        634        5    9          152       11217491
```

In the above example, line 2 shows that a user using account 634 made a total of 89 accesses to COS 1 and has 125 files stored in COS 1 which together total 4168147 storage units. The storage units reported by the report utility may be configured in the Accounting Policy to represent bytes, kilobytes, megabytes, or gigabytes.

Line 3 shows that 87 of the accesses to COS 1 made by account 634 were in storage class 1, and line 4 shows that 2 of them were in storage class 2. Line 3 shows that account 634 transferred 201256 bytes in or out of storage class 1 within COS 1 and line 4 shows 0 bytes in or out of storage class 2.

Site administrators may wish to write a module that will redirect the accounting data into a local accounting data base. This module would replace the default HPSS module, **acct_WriteReport()**, which writes out the HPSS accounting data to a flat text file.

Sites may also wish to correlate the accounting report with a site-provided Account Map (see section 13.2.3: *Accounting Procedures* to determine the appropriate accounting charges.

The HPSS accounting report and a copy of any current site defined accounting configuration files, if used, should be named with the date and time and stored for future reference. Site administrators may choose to write scripts to copy/archive the generated accounting report file from its original location.

13.2.3. Accounting Procedures

The amount of work needed to support accounting varies according to the style of accounting used, whether or not Account Validation is being used, and whether the site requires additional information beyond that maintained by HPSS:

- For UNIX-style accounting, with or without Account Validation, no additional work is needed to manage accounting.
- For site style accounting:
    - The site must define and maintain a mapping of user IDs to account Ids. For each user, the mapping must define the user's default account ID and any other accounts for which the user is authorized.
        - If Account Validation is enabled:
            - The user ID to account ID mapping is maintained in the Account Validation table. For each user, this table can store the default account ID of the user and any other accounts for which he is authorized. The site must maintain this table. HPSS provides the table and a tool for maintaining it, the Account Validation Metadata editor (**hpss_avaledit**). See the **hpss_avaledit** man page for more details.
            - If the site requires additional information beyond that supplied by the Account Validation table, it must create and maintain a site defined Account Map. See section 13.2.3.1:*Site Defined Accounting Configuration Files and Procedures*on page 336.
            - For Account Validation to work, a Gatekeeper must be configured. Also, the Account Validation flag must be enabled on the *Accounting Policy* window.
        - If Account Validation is disabled:

- The site will need to create a local site Account Map to maintain a list of the account IDs for which each user is authorized. This is a locally designed and written table, not supported by HPSS. See section 13.2.3.1: *Site Defined Accounting Configuration Files and Procedures* on page 336 for suggestions for designing and maintaining an Account Map.
- Administrators will need to manage user default accounts by updating the user's LDAP registry information (i.e. the "AA=<default-acct-idx>" string in the user's hpssGECOS field).

13.2.3.1. Site Defined Accounting Configuration Files and Procedures

This section describes situations in which a site would need additional tables and/or procedures beyond those provided by HPSS. It offers suggestions on ways to implement these tables and procedures.

13.2.3.1.1. Site Defined Account Maps

If a site does not use Account Validation or if the site requires additional information beyond that maintained in the Account Validation table, it will need to create a local site defined Account Map. For example, some sites may need to keep track of a department or company name. The Account Map is designed, created, and maintained by the site, not by HPSS.

The HPSS Account Index of each user should correspond to an Account Map entry that has the site information. The following are examples of possible Account Maps:

Site-style Account Map:

```
Acct       User UID  Charge          Update_flag
12         dlk  3152 5A12x401        0
27         dlk  3152 5A12x501        0
341        dlk  3152 5A12x601        0
469        dpc  1478 7A14x401        0
470        dpc  1478 7A14x501        0
471        dmb  5674 5A12x401        0
7111       dmb  5674 7A14x501        0
...        ...  ...  ...       ...
```

UNIX-style Account Map:

```
UID        Charge
1478       7A14x401
3152       5A12x401
5674       5A12x401
...        ...
```

Note that if Site-style accounting is in use and Account Validation has been enabled, the user-to-account index mappings are already maintained in the Account Validation table. If no additional mappings are needed, there is no need for the site to create a separate site defined Account Map.

13.2.3.1.2. Site Defined Account Apportionment Table

In UNIX-style accounting, the UID (as Account Index) maps only to a specific user. Some sites may wish to apportion different percentages of the charges for a single UID among different project charge codes, but without using site style accounting. HPSS does not provide a means to do this, but a site could implement its own table and utilities to do so. Here is an example of an Account Apportionment Table which a site might implement to do this:

```
UID       % of (Project(s))
1478      75(DND) 25(CBC)
3152      45(DDI) 25(DND) 30(CBC)
5674      100(DDI)
...       .................
```

The site would need to write utilities to tie the Apportionment table to the HPSS accounting report.

13.2.3.1.3. Maintaining Site Defined Accounting Files

Sites which require a site defined Account Map, Apportionment Table, or other accounting files must develop tools to maintain and modify the files according to local accounting policies. Tools will be necessary to change, add, or delete an HPSS Account Index and its associated information. Site administrators must implement their own policies on what to do when an account is deleted, a user moves to another project, or a user leaves the system.

Other needed mappings, if any, must be maintained by the site in the Account Map file. When Account Validation is disabled, the site will need to maintain the user-to-account index mappings in the Account Map file as well.

UNIX-style accounting changes of this nature are handled through the normal utilities that set up and modify users and UIDs. A basic set of site-developed utilities are described in more detail below.

- ***Add a user and account***. New entries can be made and the next available HPSS Account Index number will be assigned from the free-list. The free-list will most likely consist of the last assigned number plus one, but could include reclaimed index numbers if a site chooses to re-use Account Index numbers that were previously assigned and no longer referenced. It is not likely that a site will need to reclaim Account Index numbers, but it is an option.
- ***Delete a user***. When a user is deleted from the Account Map, the HPSS files must be reassigned to another HPSS Account Index. This should be done from the HPSS client interface side. The **update_flag** should be set to **true** to indicate that this account index number can be reclaimed. The reclaiming tool should check for files using the account number before reclaiming it. When the Account Index is reclaimed, it can be put on the free-list to be re-used. It is important to keep a copy of the Account Map that corresponds to the HPSS accounting snapshot of storage space in use for that time period so that the proper site information can be matched.
- ***Delete account***. When an account is deleted, it is handled in the same manner as when a user is deleted.
- ***Modify account***. The entries in the Account Map can be modified to correlate the Account Index to different information, but care should be taken to keep a copy of the corresponding tables for past HPSS accounting runs.

13.2.3.2. Accounting Intervals and Charges

The time between accounting runs and the charging policy for space usage should be developed after consulting the accounting requirements. The following are some guidelines to consider:

- Accounting should be run at a regular intervals, such as once per month.
- An accounting run may take several minutes, and the storage system will probably be active during the run. The resource usage reported for each user will reflect the resources used by that user at the point when the accounting run encounters that user. This is why accounting represents a blurred snapshot instead of a snapshot at a single point in time.
- Certain accounting information is kept in cache for several minutes after it has changed. For this reason, changes to a user's accounting data may not appear in an accounting report until this period of time has elapsed. Those changes which are still in cache when accounting runs will not appear on the current accounting report, but will appear on the next accounting report.
- The number of file accesses and the amount of data transferred can be taken to represent the activity level of a certain user account in the HPSS system. You may wish to charge specifically for the network and server resources consumed by this activity.
- It may be useful to charge different rates for each Class of Service. For example, a Class of Service that keeps two tape copies of each file will use up more tape cartridges than a Class of Service that keeps only a single copy of each file on tape.

Chapter 14. User Interfaces

This chapter configuration information for the user interfaces provided with HPSS for transferring files:

- Client Application Programming Interface (API)
- Parallel File Transfer Protocol (FTP) or PFTP
- HPSS Virtual File System (VFS) Interface

14.1. Client API Configuration

The following environment variables can be used to define the Client API configuration. The defaults for these variables are defined in the hpss_env_defs.h file.

HPSS_API_AUTHN_MECH specifies which authentication mechanism may be used. The values may be “krb5” or “unix”. The default is **HPSS_CLIENT_AUTHN_MECH**.

HPSS_API_BUSY_DELAY specifies the number of seconds to delay between retry attempts. Note that this value is used both for retrying initialization operations (see HPSS_API_RETRIES) and Core Server requests (See HPSS_API_BUSY_RETRIES). The default value is 15.

HPSS_API_BUSY_RETRIES specifies the number of retries to be performed when a request fails because the Core Server does not currently have an available thread to handle that request. A value of zero indicates that no retries are to be performed (i.e., the operation will be performed once), and a value of -1 indicates that retries should be attempted until either the request succeeds or fails for another reason. The default value is 3 (which will result in up to 4 attempts).

HPSS_API_DEBUG specifies whether to produce debug messages. If the Client API is compiled with debugging enabled, any non-zero value in HPSS_API_DEBUG enables debug messages. Note that non-numeric strings will be interpreted as zero. By default, these messages will go to the standard output stream (but see HPSS_API_DEBUG_PATH).

HPSS_API_DEBUG_PATH specifies the destination of debug messages; default is “stdout”. If HPSS_API_DEBUG_PATH is set to "stdout" or "stderr", debug messages will be written to the standard output or standard error I/O streams, respectively. Otherwise, the value will be interpreted as the pathname of a file. The Client API must be compiled with debugging enabled.

HPSS_API_DESC_NAME specifies the descriptive name used in HPSS log messages if the logging feature of the Client API is enabled. The default value is “Client Application”.

HPSS_API_DISABLE_CROSS_REALM specifies cross-realm authentication. When cross-realm authentication is disabled, a client will not be allowed to access directories which are located in another security realm. The default value is zero (meaning that cross-cell authentication is enabled). To disable cross realm authentication, HPSS_API_DISABLE_CROSS_REALM must be set to a numeric, non-zero value.

HPSS_API_DISABLE_JUNCTIONS specifies whether junction traversal is enabled. When junction traversal is disabled, a client will not be allowed to access directories which require traversal to the directory via a junction. The default value is zero (meaning junction traversal is enabled). To disable junction traversal, HPSS_API_DISABLE_JUNCTIONS must be set to a numeric, non-zero value.

HPSS_API_DMAP_WRITE_UPDATES specifies the frequency of cache invalidates that are issued to the DMAPI file system while writing to a file that is mirror in HPSS. The value indicates the number of write operations between cache invalidates. The default value is 20.

HPSS_API_HOSTNAME specifies the hostname to be used for TCP/IP listen ports created by the Client API. The default value is HPSS_HOST. This value can have a significant impact on data transfer performance for data transfers that are handled by the Client API (i.e., those that use the **hpss_Read** and **hpss_Write** interfaces). The default is **HPSS_HOST**.

HPSS_API_LIMITED_RETRIES specifies the number of retry attempts before a limited retry error operation fails. Currently, no operations are defined as "limited retry error operations". The default value is 1.

HPSS_API_MAX_CONN specifies the number of connections that are supported by the Client API within a single client process. If HPSS_API_MAX_CONN is set to zero, the number of connections is equal to the default supported by the HPSS connection management software - currently 20. If HPSS_API_MAX_CONN is nonzero, it is the number of connections to be used.

HPSS_API_RETRIES specifies the number of retries to attempt when an operation fails. Currently this class of operation includes library initialization and communications failures. A value of zero indicates that no retries are to be performed (i.e., the operation will be attempted once), and value of -1 indicates that the operation will be retried until successful. The default value is 4 (meaning the operation will be attempted up to five times).

HPSS_API_RETRY_STAGE_INP specifies whether retries are attempted on attempts to open files in a Class of Service that is configured for background staging on open. A numeric, non-zero value (the default) indicates that opens which would return **-EINPROGRESS** to indicate that the file is being staged will be retried (using HPSS_API_RETRIES and HPSS_API_BUSY_DELAY to control the number and timing of retries), while a value of zero indicates that the **-EINPROGRESS** return code will be returned to the client. The default value is "1".

HPSS_API_REUSE_CONNECTIONS specifies whether TCP/IP connections are to be left open as long as a file is open or are to be closed after each read or write request. A numeric, non-zero value will cause connections to remain open, while a value of zero will cause connections to be closed after each file access. The default value is zero.

HPSS_API_RPC_PROT_LEVEL specifies the RPC protection level. Valid values are “connect”, “packet”, “packet integrity”, and “packet privacy”. The default is **HPSS_RPC_PROT_LEVEL**.

HPSS_API_SAN_3P specifies whether to enable SAN3P transfers. Valid values are “on” an d “off”. The default is on.

HPSS_API_SITE_NAME specifies the site name to be used when initializing the HPSS security services. The default value is **HPSS_SITE_NAME** .

HPSS_API_TOTAL_DELAY specifies the number of seconds to continue retrying requests. A value of zero indicates that no there is no time limit. The default value is zero.

HPSS_API_USE_PORT_RANGE specifies whether the HPSS Mover(s) should use the configured port range when making TCP/IP connections for read and write requests. A numeric, non- zero value will cause Movers to use the port range, while a value of zero will cause Movers to allow the operating system to select the port number. The default value is zero.

HPSS_KRB5_KEYTAB_FILE specifies the name of the file containing the security keys necessary for successfully initializing the Client API for Kerberos authentication. The default is auth_keytab:/var/hpss/

etc/hpss.keytab.

HPSS_UNIX_KEYTAB_FILE specifies the name of the file containing the security keys necessary for successfully initializing the Client API for UNIX authentication. The default is auth_keytab:/var/hpss/etc/hpss.unix.keytab.

14.2. FTP/PFTP Daemon Configuration

The pftp_client binary is an enhanced ftp client providing parallel data transfer facilities, HPSS specific features, and transfer of files greater than 4GB. The HPSS binary, hpss_pftpd, is an enhanced FTP Server facilitating parallel transfers into HPSS. This binary ONLY functions with HPSS. If you are using an "ftp" binary or a "pftp_client" binary to access HPSS the site MUST run the hpss_pftpd binary. The hpss_pftpd binary provides FTP services for both the standard system FTP client and the parallel-enhanced HPSS pftp_client. **NOTE**: NOT every feature of every system FTP client will be honored by the hpss_pftpd. Additionally, system FTP clients cannot take advantage of the parallel transfer features nor the increased file size capacity provided by the hpss_pftpd. If a site wishes to provide FTP services to both the system FTP Server, ftpd, and HPSS, hpss_pftpd, it will be necessary to set up the two services to listen on different ports.

There are four steps that must be performed prior to using the FTP user interface to transfer HPSS data:

1. Verifying the FTP Initial Configuration
2. Configuring the FTP Daemon Syslog
3. Defining the FTP Access
4. Creating FTP Users

These steps are described in more detail in the paragraphs that follow.

Step 1. Verifying the FTP Configuration

During the HPSS infrastructure configuration phase, the following files were created or modified by the **mkhpss** script:

- **/etc/services:** Verify that the **hpssftp** and **hpssftp-data** entries were added correctly. Ports 20 and 21 are the ports of choice to prevent having to educate every user to alternate ports. By default, **mkhpss** uses ports 4020/4021. It may be necessary to move the existing ftp and ftp-data to other ports when assigning the hpssftp and hpssftp-data to ports 20/21.
- **/etc/inetd.conf or /etc/xinetd.conf:** Verify that the **hpssftp** entry was added correctly. The only flag that has relevance is the **-cFileName** flag which specifies an alternative **HPSS.conf** file (located in **/var/hpss/etc**).

Refresh inetd or restart xinetd after making these changes:

```
% refresh -s inetd            # (AIX only)
% /etc/init.d/xinetd restart # (Linux only)
```

A “TCP Wrapper” application may be used for initiating the HPSS PFTP Daemon, to make it more secure, however such a program is not provided by the HPSS project.

The file **/var/hpss/etc/HPSS.conf** provides the options for both an HPSS Parallel FTP Daemon and a non-HPSS Parallel FTP Daemon (DIS^{2}COM PFTP Daemon). This file should be customized as needed. Refer to the HPSS.conf man page or the *HPSS Installation Guide*, Appendix D for details. NOTE: it may be necessary for a site to merge older copies of the HPSS.conf file with the template if modifications have been made since the prior release. There is no conversion job to perform this task.

Step 2. Configuring the FTP/PFTP Daemon Syslog

The PFTP Daemon attempts to write to the system log (using **syslog()**). To enable this output, follow local system procedures for capturing this information. The PFTP Daemon writes to the **LOG_DAEMON** syslog facility.

Step 3. Defining the PFTP Access

In the following statements, **{FTPBaseDir}** defaults to "/var/hpss" and **{ftpaccess}** defaults to "ftpaccess". These values may be changed in the HPSS.conf file. The **{FTPBaseDir}/etc/{ftpaccess}** file defines access, options, and restrictions for the server. An ftpaccess.tmpl file is provided in the **$HPSS_ROOT/config/templates** directory for the HPSS Administrator to customize appropriately. The pound sign (#) at the beginning of a line indicates a comment. Options are enabled by removing the comment symbol (#) and conversely disabled by introducing this symbol. Some options are specific to individual machines, so each HPSS Parallel FTP Daemon should read its own {**ftpaccess**} file.

The access options can be defined as follows:

```
# Define a class of users.
# class         <class> { anonymous | guest | real } <user>
[ <user> ... ]
#
# <user> is a full network address, including wildcarding, e.g.,
*.nsl.nersc.gov.
class hpss_class real *
class hpss_anon anonymous *
#
# Define a limit on the number of users from a certain class
# that can be using the ftp server at specific times.
# limit         <class> <n> [<times>] [<msg_file>]
#
# The <times> clause has the following format:
#       <day><time>-<time>|<day><time>-<time>|... (no white space!)
#               or "Any"
#       where <day> is one of "Su", "Mo", "Tu", "We", "Th", "Fr",
#                             "Sa", "Wk"
#                   "Wk" matches any day of the week
#             <time> is the time in 24-hour format
#       "Any" matches any day and time
#             (used to specify <msg_file> for all days and times)
# limit hpss_class 1 Tu0600-1200
limit hpss_class 3
#
# Display a banner prior to the user prompt (before log in.) Very
# useful for informing the user that he/she is dealing with the
# HPSS file systems rather than the local file system of the
# host executing the FTP Daemon. This is highly recommended!
banner <pathname/filename>

# Control for logging (sent to syslog()).
log [ commands ]  [ anonymous ]  [ { inbound } ]
[ transfers ]  [ guest ] [ { outbound } ] [ real ] [debug]

# Set the maximum number of login attempts before closing the
# connection:
loginfails  <login_failure_threshold>

# Determine the appropriate behavior when needed data must be staged.
# <stage-code> can have several values:
# -1 wait forever
# 0 do not wait (default) - inform user that data needs staging and
#                           exit
# >0 wait for n seconds - if staging takes more than n seconds, inform
#                         the user and exit
# note: This option requires that the relevant storage class be
# configured in a compatible way, or behavior may not be as
# expected. For example, setting wait_on_stage to -1 (wait forever)
# for a SC with no-stage set would make little sense.
wait_on_stage <stage-code>
#
# Define users as being guests (which means they cannot
# cd out of their own subdirectory).
guestgroup <group> [ <group> ... ]

# Deny access to a set of users, <addrglob> is full network address,
# including wildcarding, e.g., *.nsl.nersc.gov.
deny  <addrglob> [ <msg_file> ]

# Send shutdown message to current FTP users.
shutdown <shutdown_info_pathname>

# Send message to users, possible on access to a directory.
message <pathname> [ <when> ]

# Display prompt to read a file, possible on access to a directory.
readme <pathname> [ <when> ]
# The <when> clauses may be either "login" (in which case the file
# is displayed when the user logs in) - Applicable only for anonymous
# or "cwd=[<directory_name> | *]",
# (in which case the file is displayed when the user changes to the
# named directory, with '*' wildcarding all directories).
# ONLY displays message on FIRST entrance.  Directory MUST be full
path
```

- **Message/readme/banner/shutdown** (message lines) are text files, with the following keywords (all one character in length) recognized, identified by a preceding **%**:

Table 5. Banner Keywords

Keyword	Description
M	Class limit
T	Current time on FTP Daemon
C	Current working directory
R	Remote hostname
L	Local hostname
U	User name
s	Shutdown time
d	User disconnect time
r	Connection deny time
%	%

- The format of the **<shutdown_info_pathname>** file is:

```
<year> <mon> <day> <hour> <min> <deny> <disc>
```

Message lines contain keywords mentioned above. For example:

```
1994 1 28 16 0 120 30
System shutdown at %s (current time is %T)
New FTP sessions denied at %r
FTP users disconnected at %d
```

indicates that shutdown will be 1/28/94 at 16:00, with users disconnected at 15:30 and sessions denied at 14:40 (the 120 indicates 1 hour, 20 minutes).

- The **<when>** clauses may be either **login** (in which case the file is displayed when the user logs in) or **cwd=[<directory_name> | *]** (in which case the file is displayed when the user changes to the named directory, with * wildcarding all directories).
- The **<times>** clause has the following format:

<day><times>-<time> | <day><time>-<time> | . . .

where **<day>** is one of **Su, Mo, Tu, We, Th, Fr, Sa, Su, Wk**, or **Any** and **<time>** is the time of day on a 24-hour clock. For example, to specify Tuesday between 8:00 am and 4:00 pm:

Tu0800-1600

The **Wk** entry for the **<day>** clause indicates a week day (Monday-Friday), and **Any** indicates all days of the week.

Step 4. Creating FTP Users

In order for an HPSS user to use FTP, a UNIX and/or Kerberos userid and password must be created. Refer to Section 3.3.2.1: *The hpssuser Utility* on page 35 for information on how to use the hpssuser utility to create the userid and password and set up the necessary configuration for the user to use FTP. Note that this step should not be done until the Core Server is running so that the hpssuser utility can create the home directory for the FTP user.

The /**opt/hpss/bin/hpss_pftppw** utility can be used to set the encrypted passwords in the **/var/hpss/etc/passwd** file. The syntax for this utility is as follows:

hpss_pftppw <userid> [<password file pathname>]

The utility will prompt the user for the old and new passwords. If root has to change the password it is necessary to manually edit the file and remove the existing encrypted password. The password file pathname argument can be used to specify a password file other than the default file, **/var/hpss/etc/passwd**.

To enable anonymous FTP, the "**hpssftp**" user must be defined in either the HPSS FTP password file or in the Kerberos KDC and LDAP registry (depending on which authentication mechanism is enabled). In addition, the entry for the "**hpssftp**" user must contain a home directory defined to be a non-NULL value. Anonymous FTP users will have the "root" directory set to "**/anonymous**" to prevent anonymous users from accessing other portions of HPSS. This means that for anonymous access the directory **/anonymous** MUST exist!

To disable anonymous FTP, either:

1. Remove the hpss_anon class from the ftpaccess file

   - and/or -

2. Add **hpss_ftp**, **anonymous** or **guest** to the HPSS FTP user file (normally "**/var/hpss/etc/ftpusers**").

   Security Policies and a knowledge of the security requirements should be carefully examined and thoroughly understood before allowing "anonymous" FTP.

14.3. HPSS VFS Interface Configuration

This section provides an overview of the HPSS VFS Interface. It describes the process to prepare the environment, install the VFS Interface software, mount and unmount file systems, and discusses the utilities available to manage the interface.

14.3.1. HPSS VFS Interface Overview

The VFS Interface package consists of the following components:

- HPSS VFS Kernel Module (hpssfs.ko) – The VFS Interface's Linux loadable module consists of two parts. A plug-in to the bottom of the Linux VFS (Virtual File Switch) abstraction layer and a character device (/dev/hpssfs0) that is used to communicate with the HPSS VFS Daemon (hpssfsd). All POSIX I/O system calls like open, read, write, close, etc. are first handled by the VFS abstraction layer after which they are passed down to appropriate functions in the Kernel Module. The Kernel Module translates the POSIX I/O requests into HPSS requests and then forwards these requests (via /dev/hpssfs0) to the HPSS VFS Daemon (hpssfsd), which in turn sends requests to the Core Server.

- HPSS VFS Device Special File (/dev/hpssfs0) – This character device special file corresponds to a Linux character device and provides the means for sending HPSS requests and receiving HPSS replies between the Kernel Module and the HPSS VFS Daemon.
- HPSS VFS Daemons (hpssfsd) – The daemon is linked against the HPSS Client API library. It translates Kernel Module requests to Client API library calls and converts Client API library call responses back into Kernel Module replies. Instances of the HPSS VFS Daemon are started for every Linux file system mount.
- HPSS VFS Mount Helper (mount.hpssfs) – The mount helper is spawned by the mount command.
- The Mount Helper calls the mount system call which in turn drills down to the Kernel Module to do kernel mount processing. The mount helper also spawns one or more HPSS VFS Daemons.

The installation and configuration process populates the system with the above components.

14.3.2. Supported Linux Versions

The VFS Interface is currently only supported on Linux platforms for the Intel x86 and Power PC with the following kernel version:

- 2.6.18-164.EL5smp (RHEL 5.4 on Intel x86 64-bit and Power PC)

Use the "uname -a" command to determine the kernel version and architecture of the client machine. It is necessary to modify the client machine's operating system to match the above prerequisites since the operating system level must match the environment in which the VFS Interface rpm packages were generated.

14.3.3. Installation and Configuration of VFS

14.3.3.1. Extracting from HPSS Source Tree

In additional to the VFS specific code, the HPSS client API is a pre-requisite for installing VFS on a client machine. For that reason, the extraction process includes both the ‘fs’ (VFS specific code) and ‘clnt’ (HPSS Client API) packages.

On the HPSS Core Server machine as ‘root’:

```
% mkdir /tmp/vfs_client
% cd /opt/hpss
% make BUILD_ROOT=/tmp/vfs_client build-clnt build-fs
```

Create a tar file that can be used to build the client code on the VFS machine:

```
% cd /tmp/vfs_client
% tar -cvf ../vfs_client.tar *
```

Copy (i.e. scp) the new tar file to the client machine.

On the client machine, untar and build the client tree.

```
% cd /opt/hpss
% tar -xvf <path of tar file>vfs_client.tar
```

Update Makefile.macros if needed. If compiling in 64-bit mode, make sure that:

```
BIT64_SUPPORT   = on
USE_PTHREAD_25  = on
```

14.3.3.2. Compiling/Building

As ‘root’, ‘cd’ to ‘/opt/hpss’, and then compile the HPSS Client API source code. Do not include the ‘fs’ package in this step, that is complied separately.

```
% make clnt
```

Now build VFS in the module source directory.

```
% cd /opt/hpss/src/fs/linux2.6
```

Build the kernel module:

```
% make
```

Install the kernel module & headers and update module dependencies:

```
% make install
```

14.3.3.3. Modifying the Kernel

Load the kernel module by following these steps:

IMPORTANT: If VFS already exists on the machine, un-mount all vfs file systems first!

```
% rmmod hpssfs  # this ensures that there isn't a pre-existing
  module loaded
% modprobe hpssfs
% make config
% MAKEDEV hpssfs
% /sbin/chkconfig –add hpssfs
```

Build and install the application daemon by following the instructions given by the build-help target of the makefile in the directory above this one.

```
% cd /opt/hpss/src/fs
```

Follow the instructions reported by issuing:

```
% make build-help
```

Build the HPSS VFS Daemon.

```
% make
```

Install the HPSS VFS Daemon.

```
% make install
```

When complete, verify that the following VFS interface components are installed in the following locations:

- HPSS VFS Kernel Module: /lib/modules/<linux_version>/kernel/fs/hpssfs/hpssfs.ko
- HPSS VFS Device Special File: /dev/hpssfs0
- HPSS VFS Daemon: /sbin/hpssfsd
- HPSS VFS Mount Helper: /sbin/mount.hpssfs

14.3.3.4. Client API - Pre-Requisite

VFS is dependent upon the HPSS Client API being installed, and configured, first. It is recommended that the HPSS Client API be installed/configured, then tested using an API based tool/program like 'scrub', example code (as supplied by the deployment/support team), or other HPSS Client API based applications before attempting to install/use VFS itself. Verifying that the HPSS Client API is working correctly will eliminate it as a possible problem if VFS troubleshooting becomes necessary.

Create the following directories on the VFS machine:

```
% mkdir /var/hpss/etc
% mkdir /var/hpss/cred
% mkdir /var/hpss/tmp
```

On the Core Server machine, use mkhpss to create the client config bundle:

```
% mkhpss
```

Select "Create Config Bundle" to create a client config bundle that contains config. files from the Core Server machine:

```
[ Adding HPSS Local passwd/group/shadow files to Bundle]
[ Verifying that all files to be packaged exist ]
[ generating client config bundle in /tmp/hpss_cfg.tar ]
env.conf
ep.conf
site.conf
auth.conf
authz.conf
HPSS.conf
ieee_802_addr
core_server
mover_list
hpss.unix.keytab
passwd
shadow
group
pax: ustar vol 1, 13 files, 0 bytes read, 40960 bytes written.
## run command exited with status 0
```

Untar the client config bundle, generated on the Core Server machine, under the /var/hpss/etc directory.

- Make sure that all the appropriate /var/hpss/etc/*keytab files are copied to the client machine.
  - /var/hpss/etc/site.conf file should specify the Core Server machine.
  - /var/hpss/etc/core_server file must exist.
- This file is typically created on the Core Server machine using mkhpss utility after HPSS has been configured and contains these 2 lines:
  - o host: <hostname>
  - o db2svc: 60000
- Alternately, hand-create the file specifying the Core Server host name.
  - The library files specified in /var/hpss/etc/auth.conf and authz.conf are also required to authenticate the user.
  - o /opt/hpss/lib/libhpsskrb5auth.so

    - /opt/hpss/lib/libhpssunixauth.so
    - /opt/hpss/lib/libhpssldapauthz.so

- If the code is installed in a non-standard location (not /opt/hpss), update the paths in auth.conf and authz.conf to use the correct location.

If using kerberos authentication, modify the /etc/krb5.conf file on the client machine to list both Core Server in the 'realms' and 'domain_realm' sections.

```
      Example:
      default = FILE:/var/hpss/log/krb5libs.log
      kdc = FILE:/var/hpss/log/krb5kdc.log
      admin_server = FILE:/var/hpss/log/kadmind.log
      [libdefaults]
      ticket_lifetime = 24000
      default_realm = <all caps - fully qualified hostname>
      default_keytab_name = /etc/v5srvtab
      default_tkt_enctypes = des-cbc-crc
      default_tgs_enctypes = des-cbc-crc
      [realms]
      <all caps - fully qualified hostname> = {
           kdc = <fully qualified hostname>:88
           admin_server = abilene.clearlake.ibm.com:749
      }
      [domain_realm]
      <fully qualified hostname> = <all caps – fully qualified
hostname>
```

There must be a user named 'hpssfs' on the client machine and hpssfs's ‘uid’ and ‘gid’ must match the hpssfs's ‘uid’ and ‘gid’ on the Core Server machine.

Lastly, verify the client API functions properly by checking operations using an API based tool/program.

14.3.3.5. Other System Configuration Details

Check for time clock skew between the client and server machines. Sites should use the Network Time Protocol tool or equivalent utility to keep the HPSS server and client machines time synced. A time skew of 2 minutes or more will cause HPSS to refuse IO requests.

If necessary, update /etc/hosts so that the IP address of the client machine is mapped to the client hostname

```
      127.0.0.1        localhost.localdomain      localhost
      <X.X.X.X>        <fully qualified hostname> <short hostname>
```

14.4. Mounting VFS Filesystems

An HPSS fileset or directory is made available for user access by mounting it using the **mount(8)** command. The mount command accepts the mount input options directly from the command line or from the corresponding entry defined in the /etc/fstab file. By defining the mount options in the /etc/fstab file, the mount command can be issued in a much simpler fashion.

14.4.1. Mounting via the Command Line

The input arguments to the mount command include a list of utility options, a file system type, options to be passed to the Kernel Module, source device to be mounted, and the path of the mount point. The format of the mount command to mount an HPSS fileset or directory is as follows:

```
# /sbin/mount –t hpssfs <source> <target> -o <options>
```

where:

<source> can be either an HPSS fileset identifier or the pathname of the HPSS directory specified in the following format:

[LOCAL|"<site name>"]:[<path>|<fileset id>]

<target> is the local directory to mount the HPSS fileset or directory on
<options> are the mount options used to control the behavior of the mounted file system

An example of a command to mount a HPSS root directory tree is as follows:

```
# mount  -t  hpssfs  LOCAL:/  /tmnt  -o'cos=5'
```

In the above example, the "-t hpssfs" option refers to the VFS Interface filesystem type. When a mount command for a VFS Interface filesystem is issued, one or more HPSS VFS Daemons are started to manage the specified file system. In the above example, the root directory of the local HPSS system, indicated by the '***LOCAL:/***' parameter, is being mounted on the /tmnt mount point in order for users to access the HPSS files. The mount option, specified by the ***–o*** option, specifies the HPSS COS to use for file creates in the mounted HPSS fileset or directory. The complete list of supported mount options are described in the section below, Mount Options.

14.4.2. Mounting via the '/etc/fstab' File

The mount point for an HPSS fileset or directory can also be defined in the /etc/fstab file. Each line in this file specifies a mount point. An example of a /etc/fstab file with three HPSS mount entries is as follows:

```
# This file is edited by fstab-sync - see 'man fstab-sync' for details
/dev/VolGroup00/LogVol00        /                       ext3   defaults           1
1
LABEL=/boot                     /boot                   ext3   defaults         1 2
none                            /dev/pts                devpts      gid=5,mode=620    0
0
```

```
none                     /dev/shm                 tmpfs  defaults         0 0
none                     /proc                    proc   defaults         0 0
none                     /sys                     sysfs  defaults         0 0
/dev/VolGroup00/LogVol01       swap                      swap   defaults         0
0
jupiter:/auto/home       /home                           nfs    defaults         0
0
LOCAL:/                  /tmnt                    hpssfs noauto,cos=2            0
0
LOCAL:/home/user1        /home/hpss/user1         hpssfs noauto,maxsegsz  0 0
remote-site:/home/user2  /home/remote-hpss/user2  hpssfs noauto,ip=eth0          0
0
```

The last three entries are VFS Interface mount entries (indicated by the file system type "hpssfs"). The first two of these provide for the HPSS directories from the local HPSS site to be mounted on to two local mount points. The last entry provides for a directory tree from a remote HPSS site to be mounted on to a local mount point. See the man pages for the /etc/fstab file and the **mount(8)** command for more information about setting up this file.

The above HPSS directory trees then can be mounted as follows:

```
# su -
# mount /tmnt
# mount /home/hpss/user1
# mount /home/remote-hpss/user2
```
4.

14.4.3. Mount Options

The VFS Interface provides mount options to better meet specific usage scenarios. The following table lists the mount options which can be specified either via the mount command (using the -o option) or via /etc/fstab.

Table 5. HPSS VFS Interface Mount Options

Mount Option	*Mount Description*
acregtimeo	Seconds to cache file attributes. The shorter this value, the more frequently communication to HPSS is required. Synchronization between separate vfs nodes or mount points may benefit for a shorter cache period with the expense of longer latencies.
acdirtimeo	Seconds to cache directory attributes. The shorter this value, the more frequently communication to HPSS is required. Synchronization between separate vfs nodes or mount points may benefit for a shorter cache period with the expense of longer latencies.
stack	Size of stack. You should not have to change this.
debug	Doesn't do anything at this point. Possible future use.
acregtimeo	Seconds to cache file attributes. The shorter this value, the more frequently communication to HPSS is required. Synchronization between separate vfs nodes or mount points may benefit for a shorter cache period with the expense of longer latencies.

Mount Option	*Mount Description*
acdirtimeo	Seconds to cache directory attributes. The shorter this value, the more frequently communication to HPSS is required. Synchronization between separate vfs nodes or mount points may benefit for a shorter cache period with the expense of longer latencies.
flushd	The flush deamons look for data pages that need to be written to HPSS, either time to write or data pages being requested for other files. Each daemon consumes resources so care should be taken when changing this option. Another possibility is to change iomax to allow more pages to be available.
rtimeo / wtimeo	Performance gains can be realized when minimizing the number of transfers. This timeout allows multiple requests to be combined within the time limit.
rpages / wpages	Each specifies the maximum number of kernel buffer pages that can be dedicated to individual file transfers. Care must be taken to configure this correctly to the expected number of concurrent transfers and the iomax mount option.
iomax	Maximum number of kernel buffer pages (4096 bytes + overhead) that can be allocated for data buffers when transferring file data. Each mount point has its own set of data buffers. Since these pages consume RAM resources, this must be compatible with application loads and transfer goals for the system. For instance, the more pages allocated to the mount point, the less RAM available for applications. This should also be consistent with rpages and wpages.
fcsize	File Cache Size. Cache this much of the file on first access. This consumes resources so care must be taken not to make this too large. Configuring this to typical file access patterns will speed access when large enough to cache future requests.
san / nosan	Configures the mount point to transfer data from buffers directly to SAN devices.
shm	Shared Memory (need to look into this)
dio	Configures the mount point to transfer data to/from directly from application space. This bypasses Linux VFS file buffering, see ???? for a discussion on advantages that the Linux VFS buffering provides.
trace	Level of messages logged from the HPSS kernel extension to /var/log/messages. Ranges from 0 to 5 with 0 being the least and 5 being lots.
sharefd / nosharefd	
rattr	Typically short directory listings do not have to access HPSS for file attributes. Specifying this option will force all directory listings to retrieve file attributes from HPSS. If the application is always doing long directory listings this has no effect. If the application always requests short directory listings, this will delay those requests needlessly.
ip	Configure the network interfaces to use.
cos	Specify the default Class Of Service to use. This can be overridden by an ioctl() system call once the file is created but BEFORE any data is written. Standard Linux commands / library functions will not use the ioctl(), they will use the COS setting for files stored in a fileset.
ro / rw	File system will be ro=read only or rw=read write.
maxsegsz	Use the maximum segment size for the storage class. If not used, the minimum segment size is used. An application can use ioctl() to set this per file once the file is created but BEFORE any data is written.

Mount Option	*Mount Description*
maxiowidth	maxiowidth
princ	Override for principal
auth	Override for authorization type
key	Override keytab type
keytab	Override for keytab file
stage / nostage	Default is stage. This overrides the COS setting. Application can override this by specifying O_NONBLOCK on the open() system call. An application cannot override the nostage setting. Most standard Linux commands / library functions do not specify O_NONBLOCK so this mount point option will determine what happens.
nfs	Only specify this if nfs is using this mount point.

14.4.4. Un-mounting an HPSS Filesystem

The **mount(8)** command (without command line parameters) can be used to view the list of hpssfs mounted entries. An example of a mounted hpssfs entry is as follows:

```
hpssfs(2553:2554):/ on /tmnt type
hpssfs(rw,noauto,san,dio,ip=eth2,cos=96)
```

where:

- 2553 is the HPSS VFS Daemon process ID
- 2554 is the HPSS VFS Mount Helper group ID

The **umount(8)** command is used to unmount the above mounted file system. For example:

```
# su -
# umount /tmnt
```

14.4.5. Linux 'proc' Filesystem Statistics

The VFS Interface provides a Linux proc filesystem that contains run-time statistic. The statistics can be viewed by running the 'cat' utility on various proc filesystem files.

- Show kernel module version:

```
# cat /proc/fs/hpssfs/version
```

- List all active SAN3P LUNs and their associated path:

```
# cat /proc/fs/san
```

- Show or update current trace level for a mounted file system. Valid trace levels are 1 to 5 from least to most.

```
# cat /proc/fs/hpssfs/<group ID>/trace
```

where <group ID> can be obtained using the mount utility. The trace level can be changed while the hpssfsd is running.

```
# echo 5 > /proc/fs/hpssfs/<group ID>/trace
```

- Show general information and statistics for a mounted file system:

```
# cat /proc/fs/hpssfs/<group ID>/info
```

- List files currently opened for a mounted file system:

```
# cat /proc/fs/hpssfs/<group ID>/opens
```

- List outstanding write requests for a mounted file system:

```
# cat /proc/fs/hpssfs/<group ID>/io
```

- List current requests to a daemon:

```
# cat /proc/fs/hpssfs/<group ID>/requests
```

14.5. Additional VFS Notes

14.5.1. Building an RPM Package

After setting up VFS on a machine, the admin can build a RPM package to distribute to other client machines. The HPSS Client API must still be installed and configured on the target client.

Create/edit the file 'hpssfs.spec' to contain the HPSS VFS version number

For example:

```
      Version: 6.2.1.10_%{_kversion}
```

Un-mount all VFS file systems first. Then...

```
      $ make rpm
      (newport: rpm is located in     /usr/src/redhat/RPMS/i386/hpssfs-
6.2.2.0_2.6.9-  34.ELsmp.i386.rpm)
      (rpm is located in /usr/src/redhat/RPMS/ppc64/hpssfs-
6.2.1.10_2.6.9-34.EL.ppc64.rpm)
      (Make sure version no. is correct: hpssfs-6.2.1.10...)
```

Chapter 15. Backup and Recovery

This chapter discusses the following operations:

- Backup and recover HPSS metadata
- Backup HPSS environment
- Recover HPSS user data
- Handling DB2 space shortage

15.1. HPSS Metadata Backup and Recovery

Each HPSS site is responsible for implementing and executing the HPSS metadata backup and recovery process. The HPSS administrator must ensure that HPSS metadata is backed up on a regular basis. In addition, the HPSS administrator must verify that the backup files can be used to restore the HPSS metadata up to the point of failure. Sites can choose to use a commercially available backup software tool, such as the Tivoli Storage Manager (TSM), to backup the HPSS metadata. Optionally, a set of simple and customizable backup scripts can be developed to backup the HPSS metadata. Contact your HPSS Support Representative to obtain sample backup scripts and guidance for implementing the DB2 backup process for your site.

The following sections provide an overview of the metadata backup and recovery process.

The backup topics provided in the following sections are meant only as an overview. The DB2 Data Recovery and High Availability Guide and Reference should be consulted in order to obtain a complete understanding and explanation of issues and procedures associated with enabling full recoverability of DB2 data.

15.1.1. HPSS Administrator Responsibilities for DB2

While DB2 has sophisticated backup and recovery abilities, it is fundamentally the responsibility of the HPSS administrator to ensure that these facilities are configured and working correctly. This section outlines some of those responsibilities.

While storing the DB2 data tables on reliable disk storage systems can be very useful if it becomes necessary to recover the database, it is essential that the database be backed up regularly, and that the backup media be reliable. Many sites keep multiple copies of the backup media in separate locations. It is also extremely important that site administrators verify that they can restore a failed database from the backup media. Time should be scheduled on an occasional basis to perform practice recoveries from the backup media. This could be done by creating a separate instance of the HPSS DB2 system on a separate computer and restoring it from the primary instance backup media.

HPSS databases should be backed up regularly. Most sites back up the entire database once each night, using the online backup procedure. Note that On-line backups are only available for database configurations that enable log archiving. (see section 15.1.2.1 Configuring DB2 for Online Backup on page 358). Online backups do not interfere with normal HPSS processing. It is equally important to monitor the online backup process and verify it is completing normally.

As HPSS operations take place, DB2 writes transaction log entries describing the changes that have been made to the database. These logs allow DB2 to recover all changes made to the database since the time of the last database backup, forward to the time when DB2 was stopped by a crash, hardware failure, power outage or whatever. It is vital that the DB2 log files be hosted on a highly reliable disk systems. Furthermore, DB2 log mirroring must be used to protect this information on two separate disk systems. The disks systems must also be protected using RAID 1 or RAID 5 configurations. Reliable access to the log files is more important than performance. Loss of the DB2 log files can result in a significant impact to the operation of the system, extensive downtime to repair, and potential loss of end-user data. Proper protection of the DB2 log files and associated archived backups is critical to a site's HPSS operations.

Similarly, the HPSS administrator must regularly verify that completed DB2 log files are being stored on reliable media until they are no longer needed. Many sites make multiple copies of DB2 log files and store them in separate physical locations.

In case of a catastrophic failure of the HPSS database host computer, the database can be restored to a consistent state at the point in time of the failure, if the administrator has a reliable series of backup images and log files. After the failure that caused the database failure is corrected, the HPSS databases can be restored using the backup files, and then by replaying the DB2 log files. Much of this process happens automatically when DB2 is restarted, but if you have any questions about these requirements and procedures, please contact your HPSS Support Representative.

Another responsibility of the HPSS administrator is to periodically perform RUNSTATS operations on the database tables. RUNSTATS is a DB2 process that analyzes the tables and produces statistics that are vital to DB2's performance. Without good statistics, DB2 may create very inefficient database access plans that result in very poor performance. Although frequent RUNSTATS are not necessary on HPSS databases, the administrator should have a plan to perform them on some sort of regular schedule consistent with local requirements. You should discuss this with your HPSS Support Representative.

It is also important to have a plan to perform REORGCHK operations from time to time to monitor the organization of the database tables. As a consequence of normal use, a database table may become significantly disorganized. Database performance can be enhanced by periodic reorganizations, some of which can be performed with HPSS in operation. You should also discuss this with your HPSS Support Representative.

15.1.2. Overview of the DB2 Backup Process

Without proper and systematic metadata backup, a database failure could result in total loss of the HPSS system. For this reason, backing up the DB2 databases containing HPSS metadata is essential.

DB2 databases can be backed up in one of two ways:

1. Online backups are made that can be restored to the point of failure by "rolling forward" archived logs of transaction activity since the backup.

   This is the recommended method for making HPSS backups.

2. An offline snapshot of the database data is taken at a particular point in time.

   Offline backups provide recovery only to the point in time at which they were made, and they require that there be no other activity in the database while they are being made. *Therefore, this method is not recommended for use with HPSS.*

Since HPSS installations require the use of online backups, it is useful to understand the types of backup and restore operations that can be performed with this backup type. Online backups can be described as “full”, “incremental”, or “delta”. Full backups record the full contents of the database at a point in time. Incremental backups record all the changed data since the last full backup. Delta backups record only the data that has changed since the most recent backup (of any type). Obviously, a full backup is necessary as a starting point in the recovery of a database. Incremental and delta backups help reduce the number of transaction logs that must be retrieved and applied during a restore operation.

Consider a site that keeps the following schedule for backing up a database:

```
Sunday: full
Monday: delta
Tuesday: delta
Wednesday: incremental
Thursday: delta
Friday: delta
Saturday: delta
```

Under this scheme, if the database fails between the Thursday and Friday backups, only a few backup images would be needed for a restore: Sunday's full backup is needed as a starting point. Wednesday's incremental backup includes all changes since Sunday, so it makes retrieval of Monday's and Tuesday's backups unnecessary. Beyond that, Thursday's delta backup is needed, as well as any transaction logs archived after Thursday's backup. So the needed images are: Sunday, Wednesday, Thursday, and only those logs between Thursday's backup and the point of failure.

The way DB2's backup and restore commands are implemented makes restoration simple. Basically, only the latest backup image needs to be specified. DB2 records the backup history in each backup, so it can determine which images and logs are necessary. Assuming an automated method is used for backup storage and retrieval, it can also fetch the images and restore from them. See the *DB2 Data Recovery Guide* for details.

15.1.2.1. Configuring DB2 for Online Backup

Several steps must be taken to prepare a DB2 database for online backup. Initially, a newly-created database is configured to perform offline backups. The database's configuration must be updated to indicate that subsequent backups should be made online. An initial, offline backup must then be made as a starting point for the backup process. The database will be inoperative until this initial full backup is completed. So an outline of the major steps to set up a database for online backup is:

1. Configure backup software and make certain data can be archived into it and retrieved from it.
2. Update the database configuration to enable log archiving, this will place the database in "backup pending" state.
3. Perform initial full, offline backup (database returns to normal state).
4. Schedule subsequent backups using **cron**.

To start a DB2 backup, invoke the “**db2 backup**” command line interface. Typically, the administrator would put this command into a script along with any necessary parameters to indicate the type of backup, destination, etc. The script would then be invoked from **cron**.

For details on configuring DB2 for backup, administrators should consult the *DB2 Data Recovery Guide* as well as the documentation for their backup software.

15.1.3. Overview of the DB2 Recovery Process

Upon discovering a damaged DB2 container, you must first determine the level of hardware recovery and problem determination available to you. For example, whether or not you were utilizing some level of RAID and can recover a failed disk from a good disk. The second step in recovering damaged or inaccessible DB2 containers is considering any software recovery tools available. For example, if you were utilizing backup software, such as TSM, this would enable you to replace the bad hardware and recover onto the new hardware device to a particular point in time. For details on recovering a DB2 database, see the IBM redbook titled "Data Recovery and High Availability Guide and Reference" for the appropriate version of DB2 installed on your system. This guide will help you decide how best to recover your DB2 database or damaged container.

15.2. HPSS System Environmental Backup

The HPSS administrator must also ensure that the HPSS system environment is backed up on a regular basis and to verify that the backups can be used to restore the environment. The following sections describe the environmental data that must be safeguarded.

15.2.1. HPSS Filesystem Backup

The following HPSS filesystems should be backed up using the site backup procedure:

- /opt/hpss
- /var/hpss
- /var/hpss/adm/core
- {HPSS metadata backup path}/cfg
- {HPSS metadata backup path}/subsys1
- /db2/backups/ldap (if LDAP is used)
- /var/hpss/hpssdb
- /var/hpss/hpssdb/ldap (if LDAP is used)
- {DB2 log path}/cfg
- {DB2 log path}/subsys1
- {DB2 log path}/ldap (if LDAP is used)
- {DB2 log archive path}/cfg
- {DB2 log archive path}/subsys1
- {DB2 mirror log path}/cfg
- {DB2 mirror log path}/subsys1
- {DB2 mirror log archive path}/cfg
- {DB2 mirror log archive path}/subsys1
- {HPSS secondary metadata backup path}/cfg

- {HPSS secondary metadata backup path}/subsys1
- /<metadata backup filesystem>
- Others site-specific filesystems

15.2.2. Operating System Backup

It is also necessary to perform appropriate OS backups on a regular basis. However, these activities are outside the scope of this document. Refer to the relevant OS documentation for the appropriate backup procedures.

15.2.3. Kerberos Backup

If Kerberos authentication is configured, periodic backups of the Kerberos infrastructure will need to be made. If mkhpss was used to configure Kerberos, the Kerberos components that must be backed up are placed in the following locations:

- The Kerberos client configuration file is /etc/krb5.conf.
- The default keytab file is /etc/v5srvtab.
- If a Key Distribution Center (KDC) is running on a node, all files related to it are placed in a single directory: /var/hpss/krb5kdc.

If you configured Kerberos manually, you'll need to determine the file locations yourself. The files that should be backed up are:

- Client configuration file
- KDC configuration file
- KDC acl_file
- KDC admin_keytab
- KDC dict_file
- KDC database_name
- KDC key_stash_file
- KDC default_keytab_name

15.3. HPSS User Data Recovery

In some situations an HPSS tape or disk volume may become damaged such that the user data stored on the volume cannot be retrieved by the HPSS servers. The following paragraphs describe how to recover the data from the damaged media.

15.3.1. Recovering HPSS Files from Damaged HPSS Volumes

Occasionally, a tape or disk volume is damaged and HPSS will experience errors while trying to read data stored on the damaged volume.

The following steps must be done to determine whether a volume is actually damaged and, if so, to prepare for and to perform the recovery process:

1. Determine the name of the potentially damaged volume.

    Attempts to read the damaged volume will result in Mover alarm messages being issued to SSM. The alarm messages will contain the name of the physical volume for which the error occurred. Record the volume name.

2. Determine if the volume is actually damaged.

    Typically, an alarm message from the Mover when it encounters a damaged volume will indicate the source of the problem. The system administrator should check the block size of the tape device on which the error occurred. HPSS tape devices should be configured with variable block sizes (see Section 7.1.2: *Enable Variable Block Sizes for Tape Devices* on page 207). When satisfied that the problems are not configuration related, the system administrator should first try to repack the suspect volume (see the **repack** man page for instructions on running **repack** from the command line). If after several attempts, all segments are not repacked, continue with the recovery procedures outlined in this section.

If the first repack attempt fails, it may be worthwhile to retry the repack several times before running the ***recover*** *utility. This is especially true if copies aren't available for the storage level that contains the suspect volumes. There are a number of intermittent or environmental errors that may cause an attempt to repack a volume (especially a tape volume) to fail, while a subsequent attempt may succeed (e.g., a failing drive, problems reading a volume in a particular drive, a drive that requires cleaning). If the suspect volume is a tape, try to get it mounted on a number of drives during the repack effort. Sometimes a tape will fail on one drive but can be read on another.*

1. Make the damaged volume completely unavailable.

    From the *HPSS Health and Status* window (Section 3.9.3: *HPSS Health and Status* on page 58), click on the Monitor menu, select the Lookup HPSS Objects and then the Cartridges and Volumes options to bring up the *Identify Cartridge or Volume* window (Section 8.5.1: *Lookup Cartridges & Volumes Window* on page 263). From this window, enter the damaged Volume Name. Then click on the CS Volume button to bring up the associated *Core Server Disk Volume* or *Core Server Tape Volume* window (Section 8.5.4.1: *Core Server Disk Volume Information Window* on page 269 and Section 8.5.4.2: *Core Server Tape Volume Information Window* on page 273). The Physical Volumes table on this window identifies all physical volumes which make up the virtual volume. These physical volumes and the data stored on them will be involved in the recovery process.

    To take a VV completely off-line as a precaution so that it will not be read, written or mounted until the data recovery process has begun, set the VV Condition to DOWN in the *Core Server Disk Volume* or *Core Server Tape Volume* window.

2. Recover the data on the damaged virtual volume.

    From the *Core Server Disk Volume* or *Core Server Tape Volume* window, set the VV Condition field to RO or EOM.

    The system administrator can now use the **recover** utility to recover HPSS data from a secondary copy (see the **recover** man page). If no secondary copy exists, the **recover** utility can be used to clean up storage resources and report what data has been lost.

The **recover** utility has several options. These options are used depending on the severity of the damage. Refer to the following sections for more information on how and when to use these options.

*The **recover** utility can only recover data from damaged volumes that are part of a copy set, and not from just any hierarchy level. The term "secondary copy" as used here means one of two things:* **1)** *If a volume is a part of a copy set in which data stored in a storage class at a higher level in the hierarchy is copied directly to each of the storage classes in the copy set. The most common example is a disk/tape/tape hierarchy in which the disk data is copied first to the first tape level, followed by the disk data being copied directly to the second tape level. The two tape storage classes are considered to be part of a copy set, and each tape volume in the copy set is or has a 'secondary copy'.* **2)** *If a tape level has a migration policy with the option 'Migrate Files' set (not 'Migrate Files and Purge'), this level and the next level form a copy set (assuming the next level does not purge), and each tape volume in the copy set is or has a 'secondary copy'.*

15.3.1.1. Recover Partially Damaged Disk or Tape Volume

In this situation, a disk or tape cartridge is damaged, and all attempts to repack the damaged volume have failed. Run **recover**, which looks for a secondary copy. If found, **recover** purges all the segments on the damaged volume and attempts to reconstruct the volume data from the secondary copy. Volumes to which **recover** can write must be available in the storage class of the damaged volume. If no secondary copy is found, **recover** will report the lost segments but will not purge the segments from the damaged volume.

15.3.1.1.1. With Secondary Copies

The **recover** utility is executed with the names of one or more HPSS physical volumes. These volumes must be in the same storage class. The virtual volumes that contain the suspect physical volumes are identified and all the storage segments on those virtual volumes are moved to other virtual volumes. All of the bitfile storage segments are purged from the primary storage level and recovered from the secondary level.

For example, to run a recovery of two volumes (VOL00100 and VOL00200) that are members of a copy storage class, enter:

```
recover VOL00100 VOL00200
```

The **recover** utility logs status messages (see the **recover** man page for status log information) as it tries to recover damaged segments. The messages should be similar to the following:

```
========= Trying to recover bitfile =========
003fcadc-53fb-10cf-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
storage level = 1 on VV = VOL00100
path ( Fileset24: /home/bill/file1)
========= Trying to recover bitfile =========
163001bc-2274-10ef-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
```

```
storage level = 1 on VV = VOL00100
path ( Fileset24: /home/bill/file2)
========= Trying to recover bitfile =========
0786ab2c-156b-1047-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
storage level = 1 on VV = VOL00100
path ( Fileset24: /home/bill/file3)
========= Trying to recover bitfile =========
.
.
.
== Setting VV condition to DOWN for empty VV VOL00100 ==
0723ab2c-176b-1047-8c81-02608c2f971f
00226b52 4631 10cf 00 00 00 17
== Setting VV condition to DOWN for empty VV VOL00200 ==
0456ab2c-176b-1047-8c81-02608c2f971f
00226b52 4631 10cf 00 00 00 17
```

At the end of the recovery, any damaged volume that has been emptied by **recover** will have its VV Condition set to DOWN. The system administrator may wish to delete resources and export the volume from the HPSS System using the Storage System Manager (SSM) after **recover** completes.

15.3.1.1.2. Without Secondary Copies

When no secondary copy of the data exists, the system administrator cannot use the **recover** utility to recover the data. In this case, the system administrator should make every attempt to repack the damaged volumes. If **recover** is used against a damaged volume with no secondary copies, any damaged segments will be reported but will not be purged from the volume. To do this, see Section 8.4.3: *Repacking and Reclaiming Volumes* on page 259. For example:

```
recover VOL00100 VOL00200
```

In the case of no secondary copies and damaged segments, status messages are logged. The messages should be similar to the following:

```
========= Can't recover bitfile =========
003fcadc-53fb-10cf-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
storage level = 1 on VV = VOL00100
path ( Fileset24: /home/bill/file1)
lost segments from this storage level
offset = 0    , length = 32768
offset = 32768, length = 32768
offset = 65536, length = 32768
.
.
========= Can't recover bitfile =========
163001bc-2274-10ef-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
```

```
storage level = 1 on VV = VOL00100
path ( Fileset24: /home/bill/file2)
lost segments from this storage level
offset = 0    , length = 32768
offset = 32768, length = 32768
offset = 65536, length = 32768
```

At the end of the recovery, no segments or volumes associated with the damaged segments are purged or deleted. About the only thing accomplished by running **recover** against a damaged storage class that is not part of a copy set is a listing of the damaged segments. This listing can also be done using the **lsvol** utility instead of **recover**, if the '-a' option is given to **lsvol**. It may be possible to use the scrub utility to copy valid data from a higher or lower level to the damaged level even if copies were not available for use by the **recover** utility. See scrub's interactive help for more information.

15.3.1.2. Cleanup Totally Damaged Disk or Tape

Imagine a situation where you have a disk over tape hierarchy, all files have been migrated to tape at some point, and one or more of your disks have been irrecoverably damaged or inadvertently reformatted. In this case you would want to clean up all storage resources that point to the damaged volume(s) without even attempting a repack. To perform this type of clean up procedure, the **recover** utility can be executed with the -x option. Running in this mode **recover** removes all storage resources associated with the specified storage class and volume. To remove all storage resources associated with volume LV000100, enter:

```
recover -x LV000100
```

The output displayed when **recover** is running in this mode should consist of notification of lost bitfile segments. The messages should be similar to the following:

```
========= Cleaning up segments for bitfile =========
003fcadc-53fb-10cf-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
storage level = 0 on VV = LV000100
path ( Fileset24: /home/bill/file1 )
lost segments from this storage level
this file has been migrated since last update!
========= Cleaning up segments for bitfile =========
163001bc-2274-10ef-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
storage level = 0 on VV = LV000100
path ( Fileset24: /home/bill/file2 )
lost segments from this storage level
This file has NOT been migrated since last update!
========= Cleaning up segments for bitfile =========
.
.
.
== Setting VV condition to DOWN for empty VV LD000100 ==
0723ab2c-176b-1047-8c81-02608c2f971f
```

```
00226b52 4631 10cf 00 00 00 17
```

At the end of the cleanup, all the virtual and physical volumes associated with the targeted volumes will be deleted. All the physical volumes contained in the virtual volume will be exported from the PVL. The media can then be imported back into HPSS to be reused. At this point, any of the files once residing on the damaged disk volume might be staged from tape and accessed. Notice that the output contains indications of whether each file on disk was successfully migrated to the next storage level. In the example above, **file1** was migrated successfully and can be staged back to the disk level after cleanup. The other file (**file2**) was not migrated successfully and may contain invalid data or no data at the tape level.

The **recover** utility invoked with only the **-x** option will not remove any filename entries. If a file is to be removed from HPSS entirely, one of the other tools (such as scrub; see scrub's interactive help) may be used for that purpose. Or if you wish to remove all filenames associated with damaged volume segments which have no data at any level, run the cleanup with the **-u** option (unlink) as follows:

```
recover -x -u LV000100
```

15.4. DB2 Monitoring

For the convenience of HPSS administrators, HPSS monitors several critical aspects of the underlying DB2 databases. The amount of free space in the global (often called "cfg") database is monitored by the Core Server running the lowest numbered subsystem. Similarly, the amount of free space in each of the subsystem databases (usually called subsysx, where "x" is the subsystem number) is monitored by the Core Server that runs that subsystem. The configuration settings for these monitors are found in the global and subsystem configurations. The interval at which the monitors run and the thresholds at which they send alarms can be configured to suit the HPSS administrators needs.

Starting with HPSS version 6.2.2, a DB2 Log Monitor is available in HPSS. This monitor periodically checks the DB2 transaction log files to make sure the primary and mirrored logs are congruent and apparently performing normally. It also scans the DB2 diagnostic log, *db2diag.log*, for indications of any sort of problem with the transaction logs. If any errors are reported in *db2diag.log*, the HPSS DB2 Log Monitor sends an alarm and indicates the file path and line number where the error was reported. The administrator should be able to open *db2diag.log* and examine the problem report in detail. The error is reported one time for each occurrence in db2diag.log. This monitoring function does not interfere with normal maintenance of db2diag.log. When the file is trimmed back, or removed, the monitor automatically tracks these changes.

15.5. DB2 Space Shortage

As HPSS adds files to the database, the DB2 container spaces will be consumed. If the DB2 Space Monitor is configured to monitor free space, an alarm will be generated when free space falls below a set threshold. The following sections discuss how to handle the appropriate space shortage issue.

Before performing any of the suggestions or steps listed in this section, consult the appropriate DB2 Administration Guide: Implementation for the appropriate version of DB2. It would also be wise to verify any significant changes to, or reallocation of, production DB2 resources through your IBM Houston Support Representative.

15.5.1. DMS Table Spaces

Capacity for a DMS table space is the total size of containers allocated to the table space. When a DMS table space reaches capacity (depending on the usage of the table space, 90% is a possible threshold), you should add more space to it. The database manager will automatically rebalance the tables in the DMS table space across all available containers. During rebalancing, data in the table space remains accessible.

A DB2 container may contain more than one physical disk, but a physical disk can be used in only one DB2 container. We recommend allocating the entire physical disk to a DB2 container and not leaving over any free space on the physical disk since it will be considered wasted space once the disk is allocated to the DB2 container.

The *DB2 Administration Guide for Implementation* states that you can increase the size of a DMS table space by adding one or more containers to the table space.

To add a container to a DMS table space using the Control Center:

1. Expand the object tree until you see the Table Spaces folder.
2. Right-click the table space you wish to add the container to, and select Alter from the pop-up menu.
3. Select the Containers tab, click Add, complete the information, and click OK.
4. Click Ok.

To add a container to a DMS table space using the command line, enter:

```
ALTER TABLESPACE <name> ADD (DEVICE '<path>' <size>)
```

For example:

```
ALTER TABLESPACE CFG ADD (DEVICE '/dev/rhd5' 10000)
```

Note that size is in pages based on the page size of the existing table space you are adding the container too. In this case, container refers to the DEVICE.

Note that the ALTER TABLESPACE statement allows you to change other properties of the table space that can affect performance. Refer to “Table Space Impact on Query Optimization” in the Administration Guide: Performance for more information.

15.5.2. SMS Table Spaces

In general, you cannot extend the size of an SMS table space very easily because SMS capacity depends on the space available in the filesystem and the maximum size of the file supported by the operating system. You should be able to increase the size of a filesystem using the normal operating system facilities for increasing filesystem space.

Chapter 16. Management Tools

16.1. Utility Overview

HPSS provides a variety of command-line utility programs to aid in the use and management of the system. The programs can be grouped into the following major categories:

16.1.1. Fileset and Junction Management

These programs work for HPSS filesets.

- **crtjunction** - Creates an HPSS junction.
- **deljunction** - Deletes an HPSS junction.
- **lsfilesets** - Lists all HPSS filesets.
- **lsjunctions** - Lists all HPSS junctions.

16.1.2. Tape Management

These programs manage the life cycle of tapes in the system.

- **repack** – Consolidates data on tapes by moving storage segments from sparsely populated volumes to other volumes. The emptied volumes are then available for reuse or removal from the system.
- **reclaim** – Re-initializes an empty tape for re-use after repack has emptied it, or after it has become empty through attrition. Reclaim will not re-initialize a tape that has been retired.
- **retire** - Marks a disk or tape volume as read-only and not reclaimable.
- **remove** – Deletes empty disk and tape volumes from the Core Server's metadata, then optionally exports the volumes from the PVL/PVR.
- **recover** - Salvages data from damaged volumes by reading from other copies or other levels in the hierarchy.

These tools deal with shelving tapes:

- **shelf_tape** - Selects and moves tapes from a tape library to an off-line storage area. Additionally, shelf_tape can check in tapes that previously existed on shelf.
- **shelf_tape_util** - perform various administrative functions related to shelved tapes.

16.1.3. System Info

- **dump_sspvs** - Lists the physical volumes known to a particular subsystem, including statistics about file aggregates on tapes.
- **dumpbf** - Lists the storage characteristics of a particular bitfile.
- **lsvol** - Lists the path names of bitfiles that are located on a disk or tape volume.
- **dumppv_pvl** - List the physical volumes managed by the PVL.

- **dumppv_pvr** - List the physical volumes managed by a particular PVR.
- **lshpss** - Lists various views of the system configuration.
- **lsrb** - Lists a list of bitfiles that have not been accessed since a given date/time.
- **mps_reporter** - Produces human-readable output from MPS summary files.
- **plu** - Lists files that have a purge record in HPSS metadata. Can be used to see which bitfiles have been purge-locked (e.g., bitfiles that cannot be purged).
- **lscos** - Lists all bitfile COS change records by subsystem. It also allows an administrator to cancel all COS change requests for a particular subsystem.
- **device_scan** – Lists information all SCSI-attached devices including media-changers, disks, tape.
- **scsi_home –** Displays SCSI home locations and allows SCSI home location modification.

16.1.4. System Management

- **hpssadm -** Allows administrators to perform some administration of HPSS without the graphical user interface. Not all SSM functionality is available. In particular, the program does not support all configuration operations. But it enables the administrator to manage HPSS servers, devices, storage classes, PVL jobs, and volumes, and to display alarms and events. It may be run interactively or in batch mode.
- **hpssgui** – Starts the Graphical User Interface program which is the primary administrative interface for HPSS.
- **hacl** - Allows viewing and editing of HPSS access control lists (ACLs).
- **hpss_delog** - Extracts messages from the central HPSS log and displays them in human-readable form.
- **hpssuser** - Adds, manages, and removes HPSS users.
- **rtmu** - (Real-Time Monitoring Utility) Displays the state of requests in the HPSS system.
- **hpss_avaledit** - Allows HPSS administrators at sites using Site-style accounting and Account Validation to edit the relationships between user ids, account ids, and account names.
- **hpss_server_acl** - Updates authorization data in the database.
- **hpss_krb5_keytab** - Generates a keytab in the form usable by the **hpssadm** program.
- **hpss_krb5_rmxcred** – Removes expired cached Kerberos credentials from the default credential directory.
- **hpss_unix_group** – Manages HPSS UNIX group file.
- **hpss_unix_keygen** - Create an HPSS UNIX master key.
- **hpss_unix_keytab** – A keytab generation program for the HPSS UNIX native security service.
- **hpss_unix_passwd** – Changes the HPSS UNIX native security service password for an HPSS user.

- **hpss_unix_user** - Manages HPSS UNIX password file.
- **hpss_managetables** – Creates, deletes and modifies databases, database tablespaces, tables, views and constraints. This program is normally used by **mkhpss**. It should be used by hand only by extremely knowledgeable administrators. This program can destroy database tables.
- **xmladdindex –** Creates an XML index.
- **xmldeleteindex** – Deletes an XML index.
- **schema_registrar** – Registers an XML schema with the database.
- **hpsssum** – Checksums HPSS or local files

16.1.5. User Interfaces

- **pftp_client –** Used to perform parallel file transfers between a local and remote host.

16.1.6. Testing/Debugging

- **scrub** - A general-purpose HPSS shell; a driver for the Client API library. It can be used to perform various data operations, such as reading and writing undifferentiated data, migrating and purging specific files, and browsing the HPSS namespace. Scrub also supports doing some simple operations with User-defined Attributes. It does not transfer external files into or out of HPSS.

16.1.7. Unsupported Tools

Tools in this category are not formally supported. They are provided as-is and are found in the tools/unsupported portion of the HPSS source tree.

- **disk_allocation_stat** – Calculates and displays statistics on disk usage and fragmentation for disk storage classes.
- **ftp** – (DIS^{2}COM PFTP Daemon) This is a portable version of a standard UNIX FTP daemon which supports the HPSS Parallel FTP protocols. By replacing the System FTP Daemon and configuring the DIS^{2}COM Daemon, significant improvements can be made in moving data between two UNIX-based systems. Both a Kerberized version and a non-Kerberized version are available for a number of hardware/software platforms. The HPSS Parallel FTP Client should be used to talk to this server. Do not confuse this with the HPSS FTP daemon.
- **hpssstats** - Reports HPSS statistics.
- **Load tools** - These utilities reside in tools/unsupported/src/load. Each tool allows the manual loading, editing and deletion of one or more metadata records.
- **magic** - Allows direct manipulation of namespace objects.
- **nsde** - Displays and edits Name Service database tables.
- **PVrr** – Displays the number of PV retrieval requests that have been made on a volume.
- **removesegs** – Searches for and removes storage segments that are not pointed to by any bitfile metadata. It works for both disk and tape storage segments. It requires at least two separate searches of the bitfile metadata over a period of time before it will remove orphaned segments. It

can also be used to reset the number of segments counter in disk and tape storage maps.

- **showdiskmaps** – Sends a command to the Core Server in the selected storage subsystem to dump its in-memory disk space allocation maps, then displays that information on standard output.

Appendix A. Glossary of Terms and Acronyms

ACI	Automatic Media Library Client Interface
ACL	Access Control List
ACSLS	Automated Cartridge System Library Software (Science Technology Corporation)
ADIC	Advanced Digital Information Corporation
accounting	The process of tracking system usage per user, possibly for the purposes of charging for that usage. Also, a log record message type used to log information to be used by the HPSS Accounting process. This message type is not currently used.
AIX	Advanced Interactive Executive. An operating system provided on many IBM machines.
alarm	A log record message type used to log high-level error conditions.
AML	Automated Media Library. A tape robot.
AMS	Archive Management Unit
ANSI	American National Standards Institute
API	Application Program Interface
archive	One or more interconnected storage systems of the same architecture.
attribute	When referring to a managed object, an attribute is one discrete piece of information, or set of related information, within that object.
attribute change	When referring to a managed object, an attribute change is the modification of an object attribute. This event may result in a notification being sent to SSM, if SSM is currently registered for that attribute.
audit (security)	An operation that produces lists of HPSS log messages whose record type is SECURITY. A security audit is used to provide a trail of security-relevant activity in HPSS.
bar code	An array of rectangular bars and spaces in a predetermined pattern which represent alphanumeric information in a machine readable format (e.g., UPC symbol)
BFS	HPSS Bitfile Service.
bitfile	A file stored in HPSS, represented as a logical string of bits unrestricted in size or internal structure. HPSS imposes a size limitation in 8-bit bytes based upon the maximum size in bytes that can be represented by a 64-bit unsigned integer.

bitfile segment	An internal metadata structure, not normally visible, used by the Core Server to map contiguous pieces of a bitfile to underlying storage.
Bitfile Service	Portion of the HPSS Core Server that provides a logical abstraction of bitfiles to its clients.
BMUX	Block Multiplexer Channel
bytes between tape marks	The number of data bytes that are written to a tape virtual volume before a tape mark is required on the physical media.
CAP	Cartridge Access Port
cartridge	A physical media container, such as a tape reel or cassette, capable of being mounted on and dismounted from a drive. A fixed disk is technically considered to be a cartridge because it meets this definition and can be logically mounted and dismounted.
central log	The main repository of logged messages from all HPSS servers enabled to send messages to the Log Daemon.
class	A type definition in Java. It defines a template on which objects with similar characteristics can be built, and includes variables and methods specific to the class.
Class of Service	A set of storage system characteristics used to group bitfiles with similar logical characteristics and performance requirements together. A Class of Service is supported by an underlying hierarchy of storage classes.
cluster	The unit of storage space allocation on HPSS disks. The smallest amount of disk space that can be allocated from a virtual volume is a cluster. The size of the cluster on any given disk volume is determined by the size of the smallest storage segment that will be allocated on the volume, and other factors.
configuration	The process of initializing or modifying various parameters affecting the behavior of an HPSS server or infrastructure service.
COS	Class of Service
Core Server	An HPSS server which manages the namespace and storage for an HPSS system. The Core Server manages the Name Space in which files are defined, the attributes of the files, and the storage media on which the files are stored. The Core Server is the central server of an HPSS system. Each storage sub-system uses exactly one Core Server.
daemon	A UNIX program that runs continuously in the background.
DB2	A relational database system, a product of IBM Corporation, used by HPSS to store and manage HPSS system metadata.
debug	A log record message type used to log lower-level error conditions.

DEC	Digital Equipment Corporation.
delog	The process of extraction, formatting, and outputting HPSS central log records.
deregistration	The process of disabling notification to SSM for a particular attribute change.
descriptive name	A human-readable name for an HPSS server.
device	A physical piece of hardware, usually associated with a drive, that is capable of reading or writing data.
directory	An HPSS object that can contain files, symbolic links, hard links, and other directories.
dismount	An operation in which a cartridge is either physically or logically removed from a device, rendering it unreadable and unwritable. In the case of tape cartridges, a dismount operation is a physical operation. In the case of a fixed disk unit, a dismount is a logical operation.
DNS	Domain Name Service
DOE	Department of Energy
drive	A physical piece of hardware capable of reading and/or writing mounted cartridges. The terms device and drive are often used interchangeably.
EFS	External File System
ERA	Extended Registry Attribute
ESCON	Enterprise System Connection
event	A log record message type used to log informational messages (e.g., subsystem starting, subsystem terminating). \
export	An operation in which a cartridge and its associated storage space are removed from the HPSS system Physical Volume Library. It may or may not include an eject, which is the removal of the cartridge from its Physical Volume Repository.
FDDI	Fiber Distributed Data Interface
file	An object than can be written to, read from, or both, with attributes including access permissions and type, as defined by POSIX (P1003.1-1990). HPSS supports only regular files.
file family	An attribute of an HPSS file that is used to group a set of files on a common set of tape virtual volumes.
fileset	A collection of related files that are organized into a single easily managed unit. A fileset is a disjoint directory tree that can be mounted in some other directory tree to make it accessible to users.

fileset ID	A 64-bit number that uniquely identifies a fileset.
fileset name	A name that uniquely identifies a fileset.
file system ID	A 32-bit number that uniquely identifies an aggregate.
FTP	File Transfer Protocol
Gatekeeper	An HPSS server that provides two main services: the ability to schedule the use of HPSS resources referred to as the Gatekeeping Service, and the ability to validate user accounts referred to as the Account Validation Service.
Gatekeeping Service	A registered interface in the Gatekeeper that provides a site the mechanism to create local policy on how to throttle or deny create, open and stage requests and which of these request types to monitor.
Gatekeeping Site Interface	The APIs of the gatekeeping site policy code.
Gatekeeping Site Policy	The gatekeeping shared library code written by the site to monitor and throttle create, open, and/or stage requests.
GB	Gigabyte (2^{30})
GECOS	The comment field in a UNIX password entry that can contain general information about a user, such as office or phone number.
GID	Group Identifier
GK	Gatekeeper
GSS	Generic Security Service
GUI	Graphical User Interface
HA	High Availability
HACMP	High Availability Clustered Multi-Processing - A software package used to implement high availability systems.
halt	A forced shutdown of an HPSS server.
HDM	Shorthand for HPSS/DMAP.
hierarchy	See Storage Hierarchy.
HIMF	HPSS Interim Metadata Format
HiPPI	High Performance Parallel Interface
HPSS	High Performance Storage System
HPSS-only fileset	An HPSS fileset that is not linked to an external filesystem (e.g., XFS).

IBM	International Business Machines Corporation
ID	Identifier
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
Imex	Import/Export
import	An operation in which a cartridge and its associated storage space are made available to the HPSS system. An import requires that the cartridge has been physically introduced into a Physical Volume Repository (injected). Importing the cartridge makes it known to the Physical Volume Library.
I/O	Input/Output
IOD/IOR	Input/Output Descriptor / Input/Output Reply. Structures used to send control information about data movement requests in HPSS and about the success or failure of the requests.
IP	Internet Protocol
IRIX	SGI's implementation of UNIX
junction	A mount point for an HPSS fileset.
KB	Kilobyte (210)
KDC	Key Distribution Center
LAN	Local Area Network
LANL	Los Alamos National Laboratory
LARC	Langley Research Center
latency	For tape media, the average time in seconds between the start of a read or write request and the time when the drive actually begins reading or writing the tape.
LCU	Library Control Unit
LDAP	Lightweight Directory Access Protocol
LLNL	Lawrence Livermore National Laboratory
LMCP	Library Manager Control Point
LMU	Library Management Unit

local log	An optional circular log maintained by a Log Client. The central log contains formatted messages from all enabled HPSS servers residing on the same node as the Log Client.
Location Server	An HPSS server that is used to help clients locate the appropriate Core Server and/or other HPSS server to use for a particular request.
Log Client	An HPSS server executing on each HPSS node that is responsible for sending log messages to the local log, to the Log Daemon for central logging, and to SSM to display messages in the Alarm and Event window.
Log Daemon	An HPSS server responsible for writing log messages to the central log.
log record	A record received and maintained in a central log by the HPSS Log Daemon.
log record type	An indicator of whether a message to be logged is an alarm, event, status, debug, request, security, or accounting record.
logging service	An HPSS infrastructure service consisting of a central Log Daemon, one or more Log Clients, and server-specific logging policies.
LRU	Least Recently Used
LS	Location Server
LTO	Linear Tape-Open. A half-inch open tape technology developed by IBM, HP and Seagate.
MAC	Mandatory Access Control
managed object	A programming data structure that represents an HPSS system resource. The resource can be monitored and controlled by operations on the managed object. Managed objects in HPSS are used to represent servers, drives, storage media, jobs, and other resources.
MB	Megabyte (220)
metadata	Control information about the data stored under HPSS, such as location, access times, permissions, and storage policies. Most HPSS metadata is stored in a DB2 relational database.
method	A Java function or subroutine
migrate	To copy file data from a level in the file's hierarchy onto the next lower level in the hierarchy.
Migration/Purge Server	An HPSS server responsible for supervising the placement of data in the storage hierarchies based upon site-defined migration and purge policies.
MM	Metadata Manager. A software library that provides a programming API to interface HPSS servers with the DB2 programming environment.

mount	An operation in which a cartridge is either physically or logically made readable and/or writable on a drive. In the case of tape cartridges, a mount operation is a physical operation. In the case of a fixed disk unit, a mount is a logical operation.
mount point	A place where a fileset is mounted in the XFS and/or HPSS namespaces.
Mover	An HPSS server that provides control of storage devices and data transfers within HPSS.
MPS	Migration/Purge Server
MRA	Media Recovery Archive
MSSRM	Mass Storage System Reference Model
MVR	Mover
NASA	National Aeronautics and Space Administration
Name Service	The portion of the Core Server that providesa mapping between names and machine oriented identifiers. In addition, the Name Service performs access verification and provides the Portable Operating System Interface (POSIX).
name space	The set of name-object pairs managed by the HPSS Core Server.
NERSC	National Energy Research Supercomputer Center
NLS	National Language Support
notification	A notice from one server to another about a noteworthy occurrence. HPSS notifications include notices sent from other servers to SSM of changes in managed object attributes, changes in tape mount information, and log messages that are alarm, event, and status log record message types.
NS	HPSS Name Service
NSL	National Storage Laboratory
object	See Managed Object
ONC	Open Network Computing
ORNL	Oak Ridge National Laboratory
OSF	Open Software Foundation
OS/2	Operating System (multi-tasking, single user) used on the AMU controller PC
PB	Petabyte (2^{50})
PFTP	Parallel File Transfer Protocol

physical volume	An HPSS object managed jointly by the Core Server and the Physical Volume Library that represents the portion of a virtual volume. A virtual volume may be composed of one or more physical volumes, but a physical volume may contain data from no more than one virtual volume.
Physical Volume Library	An HPSS server that manages mounts and dismounts of HPSS physical volumes.
Physical Volume Repository	An HPSS server that manages the robotic agent responsible for mounting and dismounting cartridges or interfaces with the human agent responsible for mounting and dismounting cartridges.
PIO	Parallel I/O
PIOFS	Parallel I/O File System
POSIX	Portable Operating System Interface (for computer environments)
purge	Deletion of file data from a level in the file's hierarchy after the data has been duplicated at lower levels in the hierarchy and is no longer needed at the deletion level.
purge lock	A lock applied to a bitfile which prohibits the bitfile from being purged.
PV	Physical Volume
PVL	Physical Volume Library
PVM	Physical Volume Manager
PVR	Physical Volume Repository
RAID	Redundant Array of Independent Disks
RAIT	Redundant Array of Independent Tapes
RAM	Random Access Memory
reclaim	The act of making empty tape virtual volumes available for reuse. Reclaimed tape virtual volumes are assigned a new Virtual Volume ID, but retain the rest of their previous characteristics.
registration	The process by which SSM requests notification of changes to specified attributes of a managed object.
reinitialization	An HPSS SSM administrative operation that directs an HPSS server to reread its latest configuration information, and to change its operating parameters to match that configuration, without going through a server shutdown and restart.
repack	The act of moving data from a virtual volume onto another virtual volume with the same characteristics with the intention of removing all data from the source virtual volume.

request	A log record message type used to log some action being performed by an HPSS server on behalf of a client.
RISC	Reduced Instruction Set Computer/Cycles
RMS	Removable Media Service
RPC	Remote Procedure Call
SCSI	Small Computer Systems Interface
security	A log record message type used to log security related events (e.g., authorization failures).
SGI	Silicon Graphics
shelf tape	A cartridge which has been physically removed from a tape library but whose file metadata still resides in HPSS.
shutdown	An HPSS SSM administrative operation that causes a server to stop its execution gracefully.
sink	The set of destinations to which data is sent during a data transfer (e.g., disk devices, memory buffers, network addresses).
SMC	SCSI Medium Changer
SMIT	System Management Interface Tool
SNL	Sandia National Laboratories
SOID	Storage Object ID. An internal HPSS storage object identifier that uniquely identifies a storage resource. The SOID contains an unique identifier for the object, and an unique identifier for the server that manages the object.
source	The set of origins from which data is received during a data transfer (e.g., disk devices, memory buffers, network addresses).
SP	Scalable Processor
SS	HPSS Storage Service
SSA	Serial Storage Architecture
SSM	Storage System Management
SSM session	The environment in which an SSM user interacts with the SSM System Manager to monitor and control HPSS. This environment may be the graphical user interface provided by the hpssgui program, or the command line user interface provided by the hpssadm program.

stage	To copy file data from a level in the file's hierarchy onto the top level in the hierarchy.
start-up	An HPSS SSM administrative operation that causes a server to begin execution.
status	A log record message type used to log processing results. This message type is being used to report status from the HPSS Accounting process.
STK	Storage Technology Corporation
storage class	An HPSS object used to group storage media together to provide storage for HPSS data with specific characteristics. The characteristics are both physical and logical.
storage hierarchy	An ordered collection of storage classes. The hierarchy consists of a fixed number of storage levels numbered from level 1 to the number of levels in the hierarchy, with the maximum level being limited to 5 by HPSS. Each level is associated with a specific storage class. Migration and stage commands result in data being copied between different storage levels in the hierarchy. Each Class of Service has an associated hierarchy.
storage level	The relative position of a single storage class in a storage hierarchy. For example, if a storage class is at the top of a hierarchy, the storage level is 1.
storage map	An HPSS object managed by the Core Server to keep track of allocated storage space.
storage segment	An HPSS object managed by the Core Server to provide abstract storage for a bitfile or parts of a bitfile.
Storage Service	The portion of the Core Server which provides control over a hierarchy of virtual and physical storage resources.
storage subsystem	A portion of the HPSS namespace that is managed by an independent Core Server and (optionally) Migration/Purge Server.
Storage System Management	An HPSS component that provides monitoring and control of HPSS via a windowed operator interface or command line interface.
stripe length	The number of bytes that must be written to span all the physical storage media (physical volumes) that are grouped together to form the logical storage media (virtual volume). The stripe length equals the virtual volume block size multiplied by the number of physical volumes in the stripe group (i.e., stripe width).
stripe length	The number of bytes that must be written to span all the physical storage media (physical volumes) that are grouped together to form the logical storage media (virtual volume). The stripe length equals the virtual volume block size multiplied by the number of physical volumes in the stripe group (i.e., stripe width).
stripe width	The number of physical volumes grouped together to represent a virtual volume.

System Manager	The Storage System Management (SSM) server. It communicates with all other HPSS components requiring monitoring or control. It also communicates with the SSM graphical user interface (hpssgui) and command line interface (hpssadm).
TB	Terabyte (2^{40})
TCP/IP	Transmission Control Protocol/Internet Protocol
trace	A log record message type used to record entry/exit processing paths through HPSS server software.
transaction	A programming construct that enables multiple data operations to possess the following properties: All operations commit or abort/roll-back together such that they form a single unit of work. All data modified as part of the same transaction are guaranteed to maintain a consistent state whether the transaction is aborted or committed. Data modified from one transaction are isolated from other transactions until the transaction is either committed or aborted. Once the transaction commits, all changes to data are guaranteed to be permanent.
TTY	Teletypewriter
UDA	User-defined Attribute
UDP	User Datagram Protocol
UID	User Identifier
UPC	Universal Product Code
UUID	Universal Unique Identifier
virtual volume	An HPSS object managed by the Core Server that is used to represent logical media. A virtual volume is made up of a group of physical storage media (a stripe group of physical volumes).
virtual volume block size	The size of the block of data bytes that is written to each physical volume of a striped virtual volume before switching to the next physical volume.
VV	Virtual Volume
XDSM	The Open Group's Data Storage Management standard. It defines APIs that use events to notify Data Management applications about operations on files.
XFS	A file system created by SGI available as open source for the Linux operating system.
XML	Extensible Markup Language

Appendix B. References

6. ***3580 Ultrium Tape Drive Setup, Operator and Service Guide*** GA32-0415-00
7. ***3584 UltraScalable Tape Library Planning and Operator Guide*** GA32-0408-01
8. ***3584 UltraScalable Tape Library SCSI Reference*** WB1108-00
9. ***AIX Performance Tuning Guide***
10. ***Data Storage Management (XDSM)*** API, ISBN 1-85912-190-X
11. ***HACMP for AIX, Version 4.4: Concepts and Facilities***
12. ***HACMP for AIX, Version 4.4: Planning Guide***
13. ***HACMP for AIX, Version 4.4: Installation Guide***
14. ***HACMP for AIX, Version 4.4: Administration Guide***
15. ***HACMP for AIX, Version 4.4: Troubleshooting Guide***
16. ***HPSS Error Messages Reference Manual***, November 2009, Release 6.2.
17. ***HPSS Programmer's Reference*** , November 2009, Release 6.2.
18. ***HPSS Programmer's Reference – I/O Supplement*** , November 2009, Release 6.2.
19. ***HPSS User's Guide***, November 2009, Release 6.2.
20. ***IBM 3494 Tape Library Dataserver Operator's Guide***, GA32-0280-02
21. ***IBM AIX Version 4.3 Installation Guide***, SC23-4112-01
22. ***IBM SCSI Device Drivers: Installation and User's Guide***, GC35-0154-01
23. ***IBM Ultrium Device Drivers Installation and User's Guide*** *GA32-0430-00.1*
24. ***IBM Ultrium Device Drivers Programming Reference*** WB1304-01
25. ***Interfacing Guide DAS,*** Order no. DOC F00 011
26. ***Installing, Managing, and Using the IBM AIX Parallel I/O File System***, SH34-6065-02
27. ***Parallel and ESCON Channel Tape Attachment/6000 Installation and User's Guide***, GA32-0311-02
28. ***Platform Notes: The hme FastEthernet Device Driver*** 805-4449
29. ***POSIX 1003.1-1990 Tar Standard***
30. ***Reference Guide AMU***, Order no. DOC E00 005
31. ***STK Automated Cartridge System Library Software (ACSLS) System Administrator's Guide***, PN 16716
32. ***STK Automated Cartridge System Library Software Programmer's Guide***, PN 16718

33. J. Steiner, C. Neuman, and J. Schiller*, "**Kerberos: An Authentication Service for Open Network Systems,"*** USENIX 1988 Winter Conference Proceedings (1988).

34. R.W. Watson and R.A. Coyne, "***The Parallel I/O Architecture of the High-Performance Storage System (HPSS),***" from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.

35. T.W. Tyler and D.S. Fisher, "***Using Distributed OLTP Technology in a High-Performance Storage System,***" from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.

36. J.K. Deutsch and M.R. Gary, "***Physical Volume Library Deadlock Avoidance in a Striped Media Environment,***" from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.

37. R. Grossman, X. Qin, W. Xu, H. Hulen, and T. Tyler, "***An Architecture for a Scalable, High-Performance Digital Library***," from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.

38. S. Louis and R.D. Burris, "***Management Issues for High-Performance Storage Systems***," from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.

39. D. Fisher, J. Sobolewski, and T. Tyler, "***Distributed Metadata Management in the High Performance Storage System***," from the 1st IEEE Metadata Conference, April 16-18, 1996.

Appendix C. Developer Acknowledgments

HPSS is a product of a government-industry collaboration. The project approach is based on the premise that no single company, government laboratory, or research organization has the ability to confront all of the system-level issues that must be resolved for significant advancement in high-performance storage system technology.

HPSS development was performed jointly by IBM Worldwide Government Industry, Lawrence Berkeley National Laboratory, Lawrence Livermore National Laboratory, Los Alamos National Laboratory, NASA Langley Research Center, Oak Ridge National Laboratory, and Sandia National Laboratories.

We would like to acknowledge Argonne National Laboratory, the National Center for Atmospheric Research, and Pacific Northwest Laboratory for their help with initial requirements reviews.

We also wish to acknowledge Cornell Information Technologies of Cornell University for providing assistance with naming service and transaction management evaluations and for joint developments of the Name Service.

In addition, we wish to acknowledge the many discussions, design ideas, implementation and operation experiences we have shared with colleagues at the National Storage Laboratory, the IEEE Mass Storage Systems and Technology Technical Committee, the IEEE Storage System Standards Working Group, and the storage community at large.

We also wish to acknowledge the Cornell Theory Center and the Maui High Performance Computer Center for providing a test bed for the initial HPSS release.

We also wish to acknowledge Gleicher Enterprises, LLC for the development of the HSI, HTAR and Transfer Agent client applications.

Finally, we wish to acknowledge CEA-DAM (**Commissariat à l'Energie Atomique - Centre d'Etudes de Bruyères-le-Châtel**) for providing assistance with development of NFS V3 protocol support.